Publication Number S29-677

OS/32
TEXT
USER GUIDE

PERKIN-ELMER

Computer Systems Division
2 Crescent Place
Oceanport, N J 07757

PAGE REVISION STATUS SHEET

PUBLICATION NUMBER S29-677

TITLE OS/32 TEXT User Guide

REVISION R00 DATE February 1979

PAGE	REV.	DATE	PAGE	REV.	DATE	PAGE	REV.	DATE
i/ii	R00	2/79	B-1/ B-2	R00	2/79			
iii								
iv	R00	2/79	Index -1					
ix/x	R00	2/79	thru					
			Index					
1-1			-6	R00	2/79			
thru								
1-3/								
1-4	R00	2/79						
2-1								
thru								
2-6	R00	2/79						
3-1								
thru								
3-35/								
3-36	R00	2/79						
4-1								
thru								
4-21/								
4-22	R00	2/79						
5-1								
thru								
5-43/								
5-44	R00	2/79						
6-1								
thru								
6-62	R00	2/79						
7-1								
thru								
7-4	R00	2/79						
8-1								
thru								
8-16	R00	2/79						
A-1								
thru								
A-6	R00	2/79						

LICENSING SOFTWARE
RESTRICTED RIGHTS

OS/32 TEXT is licensed software, subject to restricted rights as defined in the Department of Defense, Armed Service Procurement Regulations, ASPR, paragraph 7-104.9(a). (Rights in Data and Computer Software).

In summary, the following apply:

1. OS/32 TEXT may be used with the computer for which or with which OS/32 TEXT was acquired.

2. OS/32 TEXT may be used with a backup computer if the computer for which or with which OS/32 TEXT was acquired is temporarily inoperative.

3. OS/32 TEXT may be copied for safekeeping (archives) or backup purposes.

4. OS/32 TEXT may be modified or combined with other software, subject to the provision that those portions of the derivative software incorporating restricted rights software are subject to the same restricted rights.

   a. Provided there is no change in ownership of the computer for which or with which OS/32 TEXT was acquired, the OS/32 TEXT program may be used at any installation to which the computer may be transferred.

   b. License assignment and sublicense facilities are provided by Interdata to enable third party use of OS/32 TEXT.

   c. Restricted rights for OS/32 TEXT apply only to the following parts of the product:

      03-212 OS/32 TEXT

PREFACE

This manual describes the usage of OS/32 TEXT, Part Number 03-212. OS/32 TEXT is a program that facilitates the production and maintenance of all forms of documentation.

User Prerequisites

Users with no previous programming experience can easily learn to use OS/32 TEXT. However, prior to using OS/32 TEXT, the user must have some knowledge of OS/32 file handling, task loading, and task execution procedures, as an MTM terminal user or as a console operator. If the user is unfamiliar with these procedures, additional instruction should be obtained from the Related Documentation listed in this program.

Synopsis of Chapters in This Manual

This manual provides tutorial material that is directed to the new user, as well as reference sections that are directed to the more experienced user.

Chapters 1 and 2 contain introductory information and should be read by all users. Chapter 3 is primarily directed to the user who has no previous document formatting experience using a computer. The use of OS/32 TEXT is explained through a series of examples that demonstrate the features of OS/32 TEXT in a meaningful context. Chapter 4 which is directed more toward the experienced user discusses the capabilities provided by OS/32 TEXT. Chapters 5 and 6 are reference guides to OS/32 TEXT commands and formatting requests respectively. Chapter 7 explains the use of OS/32 TEXT-II, a version that provides only formatting capability. Chapter 8 explains error handling. Appendices contain "pull out" reference pages and task establishment information.

Related Documentation

- MTM Terminal User's Reference Manual, Publication Number 29-591, should be consulted by users operating OS/32 TEXT at an MTM terminal.

- OS/32 MT Operator's Reference Manual, Publication Number 29-574, provides information for the OS/32 system operator, and should be consulted by users operating OS/32 TEXT at the system console.

- OS/32 Edit User's Manual, Publication Number 29-576, describes the text editing commands used in OS/32 TEXT in greater detail.

System Requirements

OS/32 TEXT executes as a segmented user task under OS/32 R04-01 or higher. The system must include disc support and a suitable printing device.

OS/32 TEXT supoorts both lower and upper case character processing where upper/lower case devices are available for text entry and output.

When outputting to a line printer, OS/32 TEXT assumes that the first line available for printing is just below the paper perforation.

This manual was produced using the OS/32 TEXT program described herein.

TABLE OF CONTENTS

APPENDICES

FIGURES

CHAPTER 1
GENERAL OVERVIEW OF OS/32 TEXT OPERATION

1.1 GENERAL

This chapter describes in general terms the capabilities of OS/32 TEXT and how these capabilities are used to produce documents, letters, etc. The generation and revision of a manual such as this one, with its many revisions from inception to final version, is greatly simplified. The typist works only on those areas of text that require changes; OS/32 TEXT prints the revised document with the changes incorporated and with the pages properly arranged. Thus the often tedious cut-and-paste and retyping tasks are eliminated.

1.2 TEXT ENTRY: CREATION AND MAINTAINING TEXT

Initially the typist enters the text of the document being prepared in a manner similar to typing on a typewriter. OS/32 TEXT then retains the text which was entered so that, at any time, this text can be manipulated to perform such functions as:

- reviewing typed text
- error correction
- deletions, additions, insertions
- re-arranging of paragraphs
- printing

The text can also be uniquely identified and "filed" in the computer for future processing to perform all the functions listed above. If desired, a different file can be kept each time a document is revised.

1.3 FORMATTING: PRODUCING THE DESIRED DOCUMENT

To "format" a document the typist describes to OS/32 TEXT how the printed form should apoear. The typist does this by by inserting the appropriate formatting requests within the actual text of the document being prepared. The formatting requests are predefined words with an English meaning, such as PARAGRAPH, INDENT, etc. The formatting requests are generally embedded in the text as part of the typing operation, whenever their function is required. They can also be added, changed, deleted, etc., in the same manner as has been described for the text itself.

The formatting requests offer many options to assist the user in producing an appealing document. Some of these options are:

- definition of page dimensions and margins
- automatic page numbering
- left and right justification
- top and bottom titles
- centering text, titles
- multiple spacing

1.4 VERSIONS OF OS/32 TEXT

OS/32 TEXT is provided in two versions in the OS/32 TEXT package, Part Number S90-020:

1) OS/32 TEXT-I (file name TEXT32.TSK) is the full capability program providing all the text entry, editing and formatting features described in this manual.

2) OS/32 TEXT-II (filename TEXT32FM.TSK) is a subset of OS/32 TEXT-I providing the formatting capability without text entry and editing functions.

Installations with limited memory space may find it convenient to use OS/32 TEXT-II. In those installations where only OS/32 TEXT-II is available, editing functions are performed using a separate program, OS/32 EDIT (filename EDIT32.TSK as provided in the OS/32 package, S90-016). OS/32 EDIT provides the identical text entry and editing capabilities that are an integral part of OS/32 TEXT-I.

Throughout this manual the discussion is directed toward the primary version of OS/32 TEXT, OS/32 TEXT-I. The operation of OS/32 TEXT-II is explained in Chapter 7.

CHAPTER 2
GETTING STARTED

2.1 GENERAL

This chapter discusses the steps required to make OS/32 TEXT operational in your system: loading and starting the program are explained; interactive operation and batch operation are discussed.

If you are using an MTM system, it is assumed that you are familiar with the SIGNON/SIGNOFF procedures required to use the system. If you are not familiar with the procedures, refer to the MTM User's Reference Manual.

2.2 LOADING TEXT

In order to make OS/32 TEXT operational, the program is first loaded into computer memory from disc.

OS/32 TEXT must be an established OS task before it can be loaded. The task establishment procedure is discussed in Appendix B. In this discussion it is assumed that OS/32 TEXT resides in task form on the system disc under the name TEXT32.TSK.

Examples follow that illustrate loading OS/32 TEXT in an OS/32 or an MTM environment. In the examples, the "*" which is output by the system, indicates that the system is ready to accept a command from you, the user.

OS/32 MT Environment:

In the OS environment you enter the LOAD command followed by the TASK command. The TASK command is required to set TEXT32 as the currently selected task.

```
*LOAD TEXT32,,seg-size increment
*TASK TEXT32
```

MTM Environment:

Only the LOAD command is required.

```
*LOAD TEXT32,seg-size increment
```

"seg-size increment" in the above examples is an optional argument used to provide additional memory for TEXT32 working space, thus generally improving response time of the program. seg-size is an integer argument that represents the amount of additional space, expressed as k-bytes, allocated as the program's working space. See Appendix B for further discussion.

2.3 STARTING TEXT

After OS/32 TEXT has been loaded, the START command is used to begin execution of the program. The format of this command is the same for both the OS/32 and MTM environments. Any one of the following forms of the START command may be used to start execution of OS/32 TEXT:

```
START
START ,COMMAND=fd1
START ,LIST=fd2
START ,COMMAND=fd1,LIST=fd2
```

The simple form, START by itself, is the commonly used way of starting OS/32 TEXT. It causes OS/32 TEXT to automatically use the console device (e.g., Model 1100, Carousel) as both the command input and list output device to perform the dialog with the user. The console device (often called the terminal) is the device at which you enter commands to OS/32, MTM, or OS/32 TEXT.

If the use of the console device for both command input and output is not appropriate for your needs, one of the alternate START forms should be used to specify a different device.

2.3.1 Command Device: COMMAND = fd1

The file descriptor fd1 specifies the command input device. You enter commands to OS/32 TEXT from the command input device. If a command input device is not specified, commands are read from the console device, CON:.

2.3.2 List Device: LIST = fd2

The file descriptor fd2 specifies the device for command responses. This device is called the list device or sometimes the command output device.

If fd1 is an interactive device (CRT, Teletype, etc.) and "LIST=" is omitted from the START command, command responses are output to fd1.

If fd1 is not an interactive device, the argument "LIST=" is required.

2.4 COMMUNICATING WITH OS/32 TEXT

Once TEXT has been started with the START command TEXT displays its program identification, followed by the "greater than" (>) symbol. OS/32 TEXT outputs the > symbol whenever it expects you to enter a command:

```
OS/32 TEXT Rn
>
```

where,

- Rn is the revision number of the version of TEXT that you are using
- > is the prompt character indicating that OS/32 TEXT is ready to accept a command from the user.

OS/32 TEXT now expects you to enter a command. For example, to tell OS/32 TEXT that you want to enter text, you type the command APPEND and then depress the RETURN key. OS/32 TEXT processes your command and in this case, expects you to enter the text that you want to be formatted. When you have entered your last line of text you would indicate to OS/32 TEXT that you are finished by entering a carriage return directly following the ">" output by OS/32 TEXT. OS/32 TEXT then expects you to enter another command. At this time you may want to review what you have done so far. By entering TYPE 1- you command OS/32 TEXT to display everything you have typed from the first line to the last line. After the display of your text is complete the program is again waiting to receive a command from you.

APPEND and TYPE are examples of commands that you would use to tell OS/32 TEXT to perform certain functions. There are many more such commands to help you accomplish your task. These commands and the functions they perform for you are discussed in subsequent chapters.

2.4.1 Interactive Mode

When OS/32 TEXT is used at an interactive device such as a CRT or Carousel, OS/32 TEXT is being used in the interactive mode. In this mode, the following conditions prevail:

- the LIST argument in the START command can be omitted. If the LIST argument is omitted, command responses are output to the command input device.

- error messages are output to the command input device.

2.4.1.1 Terminal Use

Very likely, you will find that typing on most terminals is easier and quieter than using a typewriter. The keyboard layout is the same as on a typewriter except for several additional keys, some of which you will learn to use.

Transmission of data to the computer occurs whenever you depress RETURN at the end of the line you are currently typing.

Note that some terminals do not support lower case characters. This is important to consider when preparing a document.

An important advantage of a terminal over an ordinary typewriter is the ease of error correction. Methods of error correction are:

1) If you are reviewing text that has been entered into the computer, you use the editing capability in OS/32 TEXT to correct an error. Editing is performed by using the commands described in subsequent chapters.

2) If you notice an error before you transmit the current line to the computer, you can backspace to the error, correct it and retype the remainder of the line.

   Alternatively, you can delete the entire line and re-type it.

The character and line deletion keys for the Carousel, Model 1100, and Model 1200 are:

KEY	FUNCTION
CTRL X	Voids the line being typed
CTRL H or BACKSPACE	Moves the cursor back one space, causing deletion of character

2.4.2 Batch Mode

OS/32 TEXT is being used in the batch mode when OS/32 TEXT is STARTed by specifying a non-interactive device such as a card reader, as the command input device. In this mode, there is no communication with the user in a conversational manner.

Communication with OS/32 TEXT is performed indirectly by using a prepared input sequence of commands and text. It is possible to prepare all commands and text on data processing cards and store the cards on a disc file using the Spooler, or to create a disc file containing commands and text, using OS/32 EDIT or the Source Updater.

In batch mode, the following conditions prevail:

- The COMMAND argument is required in the START command
- The LIST argument is required in the START command
- Error messages are output to the list device
- All user commands are echoed to the list device
- The first error encountered causes OS/32 TEXT to terminate execution as indicated by the message:

  ```
  -END OF TASK CODE= 2
  ```

The information and explanations in this manual are primarily directed toward the interactive user. There are rarely any differences for the batch user except error handling. For specific batch user considerations for using OS/32 TEXT commands refer to the OS/32 EDIT User's Manual, Publication Number 29-576.

CHAPTER 3
INTRODUCTION TO DOCUMENT FORMATTING USING OS/32 TEXT

3.1 INTRODUCTION

This chapter presents sample sessions showing how to enter text and how to produce the desired document. Each session introduces several features of OS/32 TEXT and shows you how to use these features to prepare a document.

To obtain the maximum benefit from these sessions, you should perform the examples at a terminal. The numbers in parentheses in the left margin in the examples correspond to the explanations following each example.

General Notes:

- The examples assume that you are familiar with loading and starting OS/32 TEXT as explained in Chapter 2.
- When performing the examples at a terminal, you depress the RETURN key when you are finished typing each line. Depressing RETURN sends the line you have just typed to the computer.
- Whenever OS/32 TEXT displays the > symbol by itself OS/32 TEXT expects you to enter a command.
- Whenever OS/32 TEXT displays a number followed by the > symbol OS/32 TEXT expects you to enter a line of text or a formatting request.
- CR appears in the examples whenever you should depress RETURN and no other text is to be entered on the line.
- Formatting requests and commands are shown in the examples in upper case letters to distinguish them from the text that you enter. However, they can be entered in upper or lower case letters.

3.1.1 Example 1: Text Entry and Simple Formatting Requests

This example illustrates how to enter and format text. The steps involved are:

- Requesting OS/32 TEXT to accept text input

- Specifying a page length formatting request
- Reviewing the typed text
- Requesting formatted output

The greater than symbol (>) is a prompt from OS/32 TEXT indicating that the program is waiting for you to enter a line.

The numbers being displayed when you enter or review text are line numbers that identify each line you have typed. In subsequent examples you will learn how these numbers can be used to reference a line of text for change or error correction. The line numbers do not become part of your formatted output.

EXAMPLE 1

At the keyboard enter:

```
(1)    >APPEND
(2)     1  >.PLENGTH 23
(3)     2  >This is an example of entering text using
        3  >OS/32 TEXT.  It is very easy to use.
        4  >Let us now make a typping error
        5  >which we will
        6  >correct
        7  >in example number 2.
(4)     8  >.PARAGRAPH
        9  >You have probably noticed that we are not filling
       10  >the lines too neatly.
       11  >We want to show you that OS/32 TEXT can do this for
       12  >you - to make your job easier.
(5)    13  > CR

(6)    TYPE 1-12
(7)     1  .PLENGTH 23
        2  This is an example of entering text using
        3  OS/32 TEXT.  It is very easy to use.
        4  Let us now make a typping error
        5  which we will
        6  correct
        7  in example number 2.
        8  .PARAGRAPH
        9  You have probably noticed that we are not filling
       10  the lines too neatly.
       11  We want to show you that OS/32 TEXT can do this for
       12  you - to make your job easier.
```

(8) >FORMAT

(9) This is an example of entering text using OS/32 TEXT. It is very easy to use. Let us now make a typping error which we will correct in example number 2.

You have probably noticed that we are not filling the lines too neatly. We want to show you that OS/32 TEXT can do this for you - to make your job easier.

(10) >

EXPLANATIONS FOR EXAMPLE 1

(1) APPEND

The first command you enter when OS/32 TEXT outputs the prompt (>) is the APPEND command. This command tells OS/32 TEXT to accept lines of text from you. This command changes OS/32 TEXT from the "command mode" to the "text mode." In text mode, you can enter your text and formatting requests as desired. OS/32 TEXT remembers all the text and formatting requests that you type. Continue typing the text and formatting requests as shown in the example next to each number displayed by OS/32 TEXT.

(2) .PLENGTH 23

This formatting request sets the page length to be used for producing the formatted output of your text. Here, you set the length to 23 lines because the screen length of a CRT is 24 lines long. If you are doing these examples at a CRT and you used a line length greater than 23 lines, the formatted output of Example 1 would disappear from the screen. The standard page length normally used by OS/32 TEXT is 66 lines long, and, if the formatted text is being printed on a print device, there is no need to use the PLENGTH formatting request.

Please remember these important rules about formatting requests:

- o Formatting requests always start at the beginning of a line
- o Formatting requests always begin with a period
- o User text cannot be typed on the same line that contains a formatting request.

(3) This is an.....job easier.

Lines 2-7 and 9-12 contain text that you want to format.

(4) .PARAGRAPH

This formatting request tells OS/32 TEXT to begin a new paragraph.

(5) CR

You enter RETURN in response to a prompt to enter text, in order to change OS/32 TEXT from text mode to command mode. In command mode OS/32 TEXT is ready to accept commands from you.

(6) TYPE 1-12

You have terminated text input in (5) above and you want to review your work. TYPE tells OS/32 TEXT to display the text that you entered in lines 1-12.

(7) 1 .PLENGTH....job easier

This is the display from OS/32 TEXT in response to your command to display the text you previously entered.

(8) FORMAT

You have reviewed the text and formatting requests that you entered, and now decide to produce the formatted output. You use the FORMAT command to tell OS/32 TEXT to produce the formatted output of your text.

(9) This is....job easier.

These two paragraphs are the formatted output of your text.

(10) >

After the formatting has been completed, the prompt is output by OS/32 TEXT and you can now enter another another OS/32 TEXT command, as shown in Example 2. Example 2 uses the text that you have entered in Example 1.

3.1.2 EXAMPLE 2: Error Correction, Text Insertion and Addition

This session uses the text that you entered in Example 1 to illustrate how errors can be corrected and lines (consisting of text or format requests) can be inserted in or added to existing text. Finally, the layout of the formatted output of Example 1 is modified. The steps involved are:

- Reviewing the existing text

- Inserting formatting requests to change
  - left and right margins
  - produce double spaced output
- Correcting a typing error
- Adding new text
- Reviewing all text
- Requesting formatted output

EXAMPLE 2

At the keyboard enter:

```
(1)   >TYPE 1-
       1  .PLENGTH 23
       2  This is an example of entering text using
       3  OS/32 TEXT.  It is very easy to use.
       4  Let us now make a typping error
       5  that we will
       6  correct
       7  in example number 2.
       8  .PARAGRAPH
       9  You have probably noticed that we are not filling
      10  the lines too neatly.
      11  We want to show you that OS/32 TEXT can do this for
      12  you - to make your job easier.

(2)   >INSERT 1

(3)   1.01  >.LMARGIN 5

(4)   1.02  >.RMARGIN -5

(5)   1.03  >.SPACE 2

(6)   1.04  >CR

(7)   >TYPE 4
       4  Let us now make a typping error

(8)   >CHANGE /typping/,/typing/,4
       4  Let us now make a typing error

(9)   >APPEND
       13 >.PARAGRAPH
       14 >In example number 2 you will learn how lines
       15 >can be inserted in existing text,
       16 >how to correct a typing error from a
```

```
        17 >previous example
        18 >and how to add text at the
        19 >end of existing text.
        20 >CR

(10)    TYPE 1-
        1     .PLENGTH 23
        1.01  .LMARGIN 5
        1.02  .RMARGIN -5
        1.03  .SPACE 2
        2     This is an example of entering text using
        3     OS/32 TEXT.  It is very easy to use.
        4     Let us now make a typing error
        5     that we will
        6     correct
        7     in example number 2.
        8     .PARAGRAPH
        9     You have probably noticed that we are not filling
       10     the lines too neatly.
       11     We want to show you that OS/32 TEXT can do this for
       12     you - to make your job easier.
       13     .PARAGRAPH
       14     In example number 2 you will learn how lines
       15     can be inserted in existing text,
       16     how to correct a typing error from a
       17     previous example
       18     and how to add text at the
       19     end of existing text.

(11)    >FORMAT,HALT

(11a)   ENTER RETURN WHEN READY

(11b)   >CR

(12)    This is an example of entering text using OS/32
        TEXT.  It is very easy to use.  Let us now make a
        typing error that we will correct in example
        number 2.


        You have probably noticed that we are not filling
        the lines too neatly.  We want to show you that

(13)    >CR

(14)    OS/32 TEXT can do this for you - to make your job
        easier.
```

In example number 2 you will learn how lines can be inserted in existing text, how to correct a typing error from a previous example and how to add text at the end of existing text.

(15) >

EXPLANATIONS FOR EXAMPLE 2

(1) TYPE 1-

The TYPE 1- command tells OS/32 TEXT to display all text from the first line to the last line entered so far. If you do not enter a line number following the dash "-", the last line entered is implied.

(2) INSERT 1

This command allows you to insert text or formatting requests after the line number referenced in the command. In this case you wish to insert new lines following line 1. Notice how the inserted lines are numbered. You may continually insert lines until you tell OS/32 TEXT that you are finished entering lines. In (6) you tell OS/32 TEXT that you are finished inserting lines.

(3) .LMARGIN 5
(4) .RMARGIN -5

These are formatting requests controlling the margin settings. In this example, the standard margin settings, 12 (left) and 74 (right), are initially in effect. By using the RMARGIN and LMARGIN requests you have changed the margin settings so that the text will be formatted using 17 and 69 as left and right margin settings, respectively. LMARGIN adjusts the left margin to 17 (12+5). RMARGIN adjusts the right margin to 69 (74-5).

(5) .SPACE 2

This formatting request tells OS/32 TEXT that the formatted output is to be double spaced.

(6) CR

Entering a CR in response to a text input request from an INSERT command switches OS/32 TEXT back to command mode.

(7) TYPE 4

Here you review a line that contains an error.

(8) CHANGE /typping/,/typing/,4

To correct errors in a line you use the CHANGE command. In the command, you first enter the characters you want to change and second the replacement characters. In this case you want to change "typping" to "typing."

A slash is required to enclose the characters you want to change and the replacement characters. The slash "/" is called a delimiter. The delimiter can be any character that is not part of the character string itself. You may always use a slash unless a slash is part of the character string you want to change or a slash is part of the new string. In this case you would substitute a character of you choosing for the slash.

The number 4 is the number of the line in which the change is to be performed.

After OS/32 TEXT has changed the line, it displays the new contents of the line so that you may verify the correction.

(9) APPEND

The APPEND command was used in Example 1 to enter lines of text and formatting requests. This command always causes lines to be added at the end of existing lines.

In Example 1, no lines existed when you used APPEND. Consequently, the first line number assigned to your input line was number 1. Here, you use APPEND again. Since line 12 is the last line, the next line assigned to your input is line 13.

(10) TYPE 1-

You request OS/32 TEXT to display your text lines from line 1 to the last line.

(11) FORMAT ,HALT

You have reviewed your text and formatting requests in (10) and decided to obtain the formatted output of your text.

The comma is required to properly position "HALT" as the second argument of the command. Chapter 5 explains the use of the comma, but for the purpose of these examples you need not be concerned with its use.

HALT tells OS/32 TEXT, to stop output of your formatted text at the beginning of each page. If you do these examples on a CRT, this allows you to study each page before the next one is displayed. More importantly, if your formatted text is being printed on a device where the paper has to be fed manually, the use of HALT allows you to change the paper before each page is printed.

To tell OS/32 TEXT that you are ready for the next page to be printed, you enter RETURN.

(11a) ENTER RETURN WHEN READY

When HALT is given in the FORMAT command, this message is displayed prior to outputting the first formatted page. At this time you may want to adjust the paper used for output.

(11b) CR

You enter RETURN to tell OS/32 TEXT to output the formatted page.

(12) This is...job easier.

This is the formatted output of page 1. OS/32 TEXT waits at the end of the page, until you enter RETURN.

(13) CR

This causes OS/32 TEXT to resume output.

(14) In example...existing text.

This is the formatted output of page 2. OS/32 TEXT again waits at the end of the page.

(15) > Since there are no more pages to be output, OS/32 TEXT returns to Command Mode. Please proceed to Example 3.

3.1.3 EXAMPLE 3: Text Search, Line Deletion, Saving and Retrieving Stored Text, Exit from OS/32 TEXT

This example presents several very useful text manipulation features. You are shown how to:

- o Search the text for certain words or characters
- o Delete text
- o Save text in a disc file
- o Retrieve the saved text from the disc file
- o Terminate use of OS/32 TEXT

This session uses the text created in Examples 1 and 2.

EXAMPLE 3

At the keyboard enter:

```
(1)   >TYPE 1-
      1     .PLENGTH 23
      1.01  .LMARGIN 5
      1.02  .RMARGIN -5
      1.03  .SPACE 2
      2     This is an example of entering text using
      3     OS/32 TEXT.   It is very easy to use.
      4     Let us now make a typing error
      5     which we will
      6     correct
      7     in example number 2.
      8     .PARAGRAPH
      9     You have probably noticed that we are not filling
     10     the lines too neatly.
     11     We want to show you that OS/32 TEXT can do this for
     12     you - to make your job easier.
     13     .PARAGRAPH
     14     In example number 2 you will learn how lines
     15     can be inserted in existing text,
     16     how to correct a typing error from a
     17     previous example and
     18     how to add text at the
     19     end of existing text.
```

```
(2)     >FIND /.PARAGRAPH/,1-19
         8     .PARAGRAPH
        13    .PARAGRAPH

(2a)    >FIND =OS/32 TEXT=,1.03-
         3     OS/32 TEXT.  It is very easy to use.
        11    We want to show you that OS/32 TEXT can do this for

(3)     >DELETE 8

(4)     >DELETE 13-

(5)     >TYPE 1-
         1     .PLENGTH 23
         1.01  .LMARGIN 5
         1.02  .RMARGIN -5
         1.03  .SPACE 2
         2     This is an example of entering text using
         3     OS/32 TEXT.  It is very easy to use.
         4     Let us now make a typing error
         5     which we will
         6     correct
         7     in example number 2.
         9     You have probably noticed that we are not filling
        10     the lines too neatly.
        11     We want to show you that OS/32 TEXT can do this for
        12     you - to make your job easier.

(6)     >SAVE EXAMPLE3.RVO

(7)     >TYPE 12
         12    You - to make your job easier.

(8)     >DELETE 1-

(9)     >TYPE 1-
(10)    !NO OS/32 TEXT

(11)    >GET EXAMPLE3.RVO

(12)    >TYPE 1-
         1     .PLENGTH 23
         2     .LMARGIN 5
         3     .RMARGIN -5
         4     .SPACE 2
         5     This is an example of entering text using
         6     OS/32 TEXT.  It is very easy to use.
         7     Let us now make a typing error
         8     which we will
         9     correct
        10     in example number 2.
        11     You have probably noticed that we are not filling
        12     the lines too neatly.
        13     We want to show you that OS/32 TEXT can do this for
        14     you - to make your job easier.
```

```
(13)   >END

(14)         END OF TASK CODE=    0

(15)   *
```

EXPLANATIONS FOR EXAMPLE 3

(1) TYPE 1-

You review all the text you have typed so far.

(2) FIND/.PARAGRAPH/,1-19

FIND is the OS/32 TEXT command that allows you to search your text for the presence a specified character string. Here you want to locate the lines that contain the formatting request ".PARAGRAPH". The slashes "/" delimit the characters for which you are searching. (The delimiter can be any character but may not be part of the string it defines.) "1-19" tells OS/32 TEXT to search from line 1 through line 19.

Each time OS/32 TEXT finds a match in its search, it displays the line number and the entire text of that line.

(2a) FIND =OS/32 TEXT=,1.03-

Note that you are using a different delimiter to define the character string. Also note that spaces can be part of the string you are searching for.

"1.03-" tells OS/32 TEXT to search from line 1.03 to the end (last line) of text.

(3) DELETE 8

The DELETE command allows you to delete a single line or a group of consecutive lines. Here, you deleted line 8.

(4) DELETE 13-

Here you deleted a consecutive group of lines from line 13 to the last line.

(5) TYPE 1-

You tell OS/32 TEXT to display all text and observe that lines 8 and 13-19 have been deleted.

(6) SAVE EXAMPLE3.RVO

The SAVE command allows you to store your current text in a disc file. This text can be retrieved at a later time to continue your work.

It may be helpful at this time to discuss how the computer can "file" your work for you.

In this example you "saved" your text in a file on the disc. A disc is a magnetic device that contains one or more platters similiar to record platters. These platters are used to store text, or in fact any data you might specify. You can not only store information on these disc platters, but also read it back, or delete it, if the information is no longer required.

A 4-drawer office filing cabinet can be compared to a disc device, where each drawer represents a record platter. When you store information on a disc, the information is placed into a "file" on any one of the platters. The computer decides on which platter to place the file.

Each file has its own unique name. The computer keeps an index of these file names and the location of files on the disc. So, when you ask for a file, the computer can determine if it has the file you want and where to find it on the disc. This procedure is quite similar to the filing and retrieving of file folders in alphabetical (or similar) order in each drawer of a filing cabinet.

In the example using the SAVE command you entered "EXAMPLE3.RVO". This is the file name (also called file descriptor) under which you want the computer to file (record) your text.

Chapter 5 describes the rules on naming a file.

(7) TYPE 12

OS/32 TEXT still remembers your text after you have saved it in the disc file. The computer simply made a copy of it. Note that if you now alter this text, the disc file does not get changed.

(8) DELETE 1-

Here you request OS/32 TEXT to delete all lines. After deleting all lines OS/32 TEXT outputs a message telling you that all lines have been deleted.

(9) TYPE 1-

You wish to display all lines.

(10) !NO OS/32 TEXT

This is a message output by OS/32 TEXT in response to your request in (9). Since you deleted all lines in (8), there is no more text to display. The text exists now only in the disc file.

(11) GET EXAMPLE3.RV0

GET is the command used to read text from a disc file. Here you are retrieving the text which you saved in file EXAMPLE3.RV0.

It does not matter that you are reading the text during the same example session in which you saved it . You can do this any time. You could have stopped using OS/32 TEXT after the SAVE command, and loaded the program again then or the next day. As long as the file EXAMPLE3.RV0 exists, you can get your text back.

Please remember these important facts when using the GET command:

When you read the text file, you do not lose that file. The computer simply copies the text from the file and makes it available to OS/32 TEXT. You can now manipulate the text again as you have done in the example. The original file remains stored on this disc. You are only working on a copy of the file.

The GET command always deletes any currently existing lines of text before reading the requested file. You should always SAVE the text you were working on before entering GET. However, if you forget to SAVE your text before entering GET, OS/32 TEXT gives you a reminder message.

As the text is read from the file, line numbers starting with 1 are assigned to each line of text.

(12) TYPE 1-

Here you see that you have successfully retrieved our text. Note, also, that the lines have been renumbered.

(13) END

This command terminates the operation of OS/32 TEXT and returns control to the operating system.

If you created, changed or added any text and had not saved it, a message is displayed to remind you:

REMINDER - SAVE YOUR CURRENT OS/32 TEXT

At this time you have a second chance to save your text.

Typing a second END command terminates the operation of OS/32 TEXT. If you did not save your text, it will be lost.

(14) END OF TASK CODE= 0

This message indicates termination of OS/32 TEXT.

(15) *

The * indicates that the operating system is ready to accept a command from you.

Perhaps at this time you might want to do some of your own text manipulations using the commands and formatting requests you have been shown so far.

If you want to save some text, use the file name EXAMPLE3.RV1 or make up your own name as described in Chapter 5.

3.1.4 EXAMPLE 4: Page Layout, Page Titles and Numbers, Footnotes and Various Line Control Requests

In this example, the emphasis is on the use of formatting requests, some of which you are already familiar with from previous examples, and a number of useful new ones.

You will be shown how to adjust page margins, how to add top and bottom page titles and page numbers, and how to add a footnote in your text.

Further, you will specify various line control requests to vary the appearance of your text. And finally, you will add comments for your own use in the text of the document which you are preparing. The steps involved are:

- Inserting page layout requests
- Inserting top and bottom page title requests with page numbering
- Entering text with various line control requests
- Adding a footnote
- Using the COMMENT request to explain some of the previous steps
- Requesting formatted output of Example 4.

This example assumes that you have terminated using OS/32 TEXT in Example 3. If you have not done so, please do it now and then load and start OS/32 TEXT again as explained in Chapter 2.

EXAMPLE 4

At the keyboard enter:

```
(1)     >APPEND
(2)     1  .COMMENT   SPECIFY PAGE LAYOUT AND LINE SPACING
(3)     2  >.MARGIN 15,66
(4)     3  >.TMARGIN 3
(5)     4  >.BMARGIN 3
(6)     5  >.PLENGTH 23
(7)     6  >.SPACE 2
        7  >.COMMENT   SPECIFY TOP & BOTTOM PAGE TITLES AND NUMBERS
(8)     8  >.TTILE !OS/32 TEXT!EXAMPLE 4!PAGE %!
(9)     9  >.BTITLE // PAGE % //
        10 >.COMMENT ENTER DOCUMENT TEXT AND FORMAT REQUESTS
(10)    11 >.CENTER
(11)    12 >.UNDERSCORE
(12)    13 >Example Number 4
```

```
(13)    14 >.SKIP 2

(14)    15 >.RJUSTIFY OFF

(15)    16 >This example illustrates many of the formatting
        17 >capabilities of OS/32 TEXT.  The title "Example Number 4"
        18 >will be centered between the margin settings.
        19 >The lines of this paragraph will not be spread to
        20 >touch the right margin
        21 >because the ".RJUSTIFY" format request is OFF.
        22 >.SKIP 2

(16)    23 >.RJUSTIFY ON

(17)    24 >.LMARGIN +5

(18)    25 >.RMARGIN -5

(19)    26 >.SPACE 1
        27 >This passage will stand out in your text because it
        28 >is typed in narrower margins and single spaced
        29 >lines.
        30 >We want to illustrate the control you have to
        31 >change the spacing and margin settings.
        32 >Note that the lines are spread out to touch
        33 >the right margin because we specified ".RJUSTIFY ON".(*)
        34 >.COMMENT   RETURN TO ORIGINAL MARGIN SETTINGS AND
        35 >.COMMENT   ADD A FOOTNOTE.

(20)    36 >.LMARGIN RESET

(21)    37 >.RMARGIN RESET
(22)    38 >.FOOTNOTE BEGIN

        39 >(*) This is an example of a footnote in your text.
        40 >The footnote will be printed at the bottom of the
        41 >current page and continued on the next page
        42 >if not enough space is available.

(23)    43 >.FOOTNOTE END

(24)    44 >.PARAGRAPH +10
        25 >The ".COMMENT" formatting request used throughout example
        46 >number 4 is intended to aid you in preparing
        47 >and maintaining document text.  This feature allows
        48 >you to keep notes in the text regarding its preparation
        49 >and formatting.  The text typed on the same line
        50 >with the request
        51 >does not appear
        52 >in the formatted output of a document.
(25)    53 >CR
```

```
(26)    >TYPE 1-20
        1  .COMMENT   SPECIFY PAGE LAYOUT AND LINE SPACING
        .
        .
        .
       20 touch the right margin

(27)    >TYPE 21-42
        21 because the ".RJUSTIFY" format request is OFF.
        .
        .
        .
        42 if not enough space is available.

(28)    >TYPE 43-
        43 .FOOTNOTE END
        .
        .
        .
        52 in the formatted output of a document.

(29)    >SAVE EXAMPLE4.RVO

(30)    >FORMAT ,HALT
        ENTER RETURN WHEN READY
        >CR
```

(31) OS/32 TEXT EXAMPLE 4 PAGE 1

Example Number 4

This example illustrates many of the formatting capabilities of OS/32 TEXT. The title "Example Number 4" will be centered between the margin settings. The lines of this paragraph will not be spread to touch the right margin because the ".RJUSTIFY" format request is OFF.

PAGE 1

(32) >CR

(33) OS/32 TEXT EXAMPLE 4 PAGE 2

> This passage will stand out in your text because it is typed in narrower margins and single spaced lines. We want to illustrate the control you have to change the spacing and margin settings. Note that the lines are spread out to touch the right margin because we specified ".RJUSTIFY ON".(*)

(*) This is an example of a footnote in your text. The footnote will be printed at the bottom of the current page and continued on the next page if not enough soace is available.

PAGE 2

(34) >CR

(35) OS/32 TEXT EXAMPLE 4 PAGE 3

The ".COMMENT" formatting request used throughout example number 4 is intended to aid you in preparing and maintaining document text. This feature allows you to keep notes in the text regarding its preparation and formatting. The text typed on the same line with the request does not appear in the formatted output of a document.

PAGE 3

(36) >

EXPLANATIONS FOR EXAMPLE 4

(1) APPEND

The APPEND command tells OS/32 TEXT to accept and remember your lines of text and formatting requests.

(2) .COMMENT SPECIFY PAGE...SPACING

This special formatting request allows you to type notes for your own use, interspersed within the document text. These notes do not become part of the printed document.

(3) .MARGIN 15,66

This request sets the absolute left and right margins respectively to be used for the formatted output of the document. The numbers represent the column numbers, counted from left to right.

(4) .TMARGIN 3
(5) .BMARGIN 3

These requests tell OS/32 TEXT the number of blank lines that will appear at the top and bottom of each page.

(6) .PLENGTH 23

This request specifies the absolute number of lines per page in the formatted output. This number generally corresponds to the physical page size of the device used for printed output. Formatting requests TMARGIN and BMARGIN are offsets within these limits.

(7) .SPACE 2

This request causes double spacing of all output lines except text appearing in a footnote.

(8) .TTITLE !OS/32 TEXT!EXAMPLE 4!PAGE %!

This request specifies that three headings should be printed at the top of each page. Each heading is defined by a "!" delimiter. In the formatted output, the titles appear as follows:

- o OS/32 TEXT is left justified
- o EXAMPLE 4 is centered
- o PAGE % is right justified

The third heading contains the page numbering symbol "%". The % symbol is replaced on each page by the computed page number. OS/32 TEXT does page numbering for you.

(9) .BTITLE // Page % //

This request is the same as TTITLE in all respects except that it affects the bottom of a page. Note that you specify only the center title. (A different delimiter, "/", is entered to indicate that you have a choice of symbols.) The left and right titles are empty. The page symbol "%" is treated as described for TTITLE.

(10) .CENTER

This request causes only the next line of text to be centered between the margin settings.

(11) .UNDERSCORE

This request causes only the next line of text to be underscored. If your formatted output is directed to a CRT, the underscoring does not appear on the screen.

(12) Example Number 4

This is the "next line of text" for the requests given in (10) and (11). "Example Number 4" will be both centered and underlined.

(13) .SKIP 2

This request causes 2 blank lines to be output.

(14) .RJUSTIFY OFF

This request causes all subsequent document text lines to be left justified only.

(15) This example illustrates...request is OFF.

These are the only document text lines that are to be left justified.

(16) .RJUSTIFY ON

Here you request that right justification of document lines is resumed.

(17) .LMARGIN +5
(18) .RMARGIN -5

With these requests you narrow the margin settings from the original settings that you specified in (3). The new values are now:

	Old Value		Change		New Value
o Left margin	15	+	5	=	20
o Right margin	66	-	5	=	61

(19) .SPACE 1

This request specifies that all subsequent document text lines are to be single spaced.

(20) .LMARGIN RESET
(21) .RMARGIN RESET

These requests cause the margin settings to be restored to their values prior to their last change. The left and right margins are again 15 and 66 respectively.

(22) .FOOTNOTE BEGIN
(23) .FOOTNOTE END

These requests delimit the lines of text that are to be printed as a footnote.

(24) .PARAGRAPH +10

This request skips lines and then indents the next line of text to begin a new paragraph. The next line is indented 10 spaces and 2 blank lines appear to separate the following paragraph from the preceding text.

(25) CR

You terminate text input. OS/32 TEXT is now in command mode awaiting input of a command.

(26) TYPE 1-20
(27) TYPE 21-42
(28) TYPE 43-

Here you request OS/32 TEXT to display the lines that you have typed so far. With each TYPE command you specify the line numbers you want to review.

(29) SAVE EXAMPLE4.RV0

You save the typed text and formatting requests that you entered in a disc file named EXAMPLE4.RV0.

(30) FORMAT ,HALT

You request OS/32 TEXT to produce the formatted output of the text that you entered. OS/32 TEXT will halt at the beginning of each page until you enter a RETURN to continue.

(31) OS/32 TEXT....PAGE 1.

This is the formatted output of page 1.

(32) CR

This causes OS/32 TEXT to resume output of the formatted text.

(33) OS/32 TEXT....PAGE 2.

This is the formatted output of page 2.

(34) CR

This causes OS/32 TEXT to resume output of the formatted text.

(35) OS/32 TEXT....PAGE 3

This is the formatted output of page 3.

(36) >

Since no more pages are to be output, OS/32 TEXT returns to command mode.

3.1.5 EXAMPLE 5: Tab Setting and Table Formatting

In this example, you are shown how to set tabulation stops in OS/32 TEXT and how to produce text in table form. The steps involved are:

- o Setting tabs
- o Typing table text and associated formatting requests
- o Saving the text in a disc file
- o Requesting a formatted output

There are two ways to add text in table form in your documents: by using either the BLOCK request, or the KEEP request. These requests are used to position a table within the document you are producing.

BLOCK is used to position the table "in context." "In context" means that the table must be printed where it appears in your text. The BLOCK BEGIN and BLOCK END formatting requests delimit the text of an "in context" block. If the table does not fit on the current page, it is printed at the top of the next page and continued on subsequent pages until the entire table is printed. The remainder of the current page is left blank.

The KEEP formatting request allows the table to be printed out of context from where it appears in your text. A table printed out of context is referred to as a "floating keep." If the table does not fit on the current page, the current page is not left blank but is filled with text which appears in your document following the table. To specify a table to be printed out of context the KEEP BEGIN and KEEP END formatting requests are used to delimit the table.

This example demonstrates the use of the BLOCK request to produce the table in context within the document.

EXAMPLE 5

At the keyboard enter:

(1) `>DELETE 1-`

(2) `>OPTION TAB=;,5,30,34,36`

(3)
```
>OPTION TAB
THE TAB CHARACTER IS    ;
   5  30  34  36
```

(4)
```
>APPEND
1  >.PLENGTH 23
2  >.TMARGIN 3
3  >.BMARGIN 3
4  >This example shows how to add a table as an "In Context
5  >Block" in your text.
6  >Since the table is too long to fit on the  same page  with
7  >this text, it is printed starting at the top of the
8  >next page.
9  >.PARAGRAPH
10 >Note that the remainder of this page is blank!
```

```
      11 >.COMMENT   NOW WE TYPE THE TABLE WITH ITS FORMAT REQUESTS

(5)   12 >.BLOCK BEGIN

(6)   13 >.NOFILL

(7)   14 >.CENTER
      15 >FACTORY COSTS

(8)   16 >.LMARGIN +10

(9)   17 >.INDENT -5
      18 >ADMINISTRATIVE COSTS

(10)  19 >TO PRODUCE A;;400,000.00
      20 >TO ENGAGE SECOND SHIFT;;150,000.00
      21 >MAINTENANCE COST PER MACHINE;;;250.00
      22 >.SKIP 1
      23 >.INDENT -5
      24 >PRODUCTION COST
      25 >PRODUCT A;;;800.00
      26 >;MATERIAL;;800.00
      27 >;MACHINE HOURS;;200.00
      28 >;MAN HOURS;;200.00
      29 >.SKIP 1
      30 >.INDENT -5
      31 >LABOR COST
      32 >FIRST SHIFT (PER MAN HOUR);;;;3.00
      33 >SECOND SHIFT;;;;3.45
      34 >.SKIP 1
      35 >.INDENT -5

(11)  36 >NUMBER OF AVAILABLE MACHINES;;;      450.00

(12)  37 >.BLOCK END

(13)  38 >.LMARGIN RESET

(14)  39 >.FILL
      40 >If the table had been specified as a
      41 >"Floating Keep", this text would have been
      42 >printed on the page preceding the table!

(15)  43 >CR

(16)  >TYPE 18-36
      18 ADMINISTRATIVE COSTS
      19 TO PRODUCE A                    400,000.00
      20 TO ENGAGE SECOND SHIFT          150,000.00
      21 MAINTENANCE COST PER MACHINE        250.00
      22 .SKIP 1
      23 .INDENT -5
      24 PRODUCTION COST
      25 PRODUCT A                           800.00
      26      MATERIAL                       200.00
      28      MACHINE HOURS                  200.00
```

```
29 .SKIP 1
30 .INDENT -5
31 LABOR COST
32 FIRST SHIFT (PER MAN HOUR)          3.00
33 SECOND SHIFT                        3.45
34 .SKIP 1
35 .INDENT -5
36 NUMBER OF AVAILABLE MACHINES          450.00
```

(17) `>SAVE EXAMPLE5.RVO`

(18)
```
>FORMAT ,HALT
ENTER RETURN WHEN READY
>CR
```

(19) This example shows how to add a table as an "In Context Block" in your text. Since the table is too long to fit on the same page with this text, it is printed starting at the top of the next page.

Note that the remainder of this page is blank!

(20) `>CR`

(21)

```
                        FACTORY COSTS
     ADMINISTRATIVE COSTS
          TO PRODUCE                      400,000.00
          TO ENGAGE SECOND SHIFT          150,000.00
          MAINTENANCE COST PER MACHINE        250.00

     PRODUCTION COST
          PRODUCT A                           800.00
               MATERIAL                       800.00
               MACHINE HOURS                  200.00
               MAN HOURS                      200.00

     LABOR COST
          FIRST SHIFT (PER MAN HOUR)            3.00
          SECOND SHIFT                          3.45

     NUMBER OF MACHINES AVAILABLE             450.00

>CR
```

(22) If the table had been specified as a "Floating Keep", this text would have been printed on the page preceding the table!

(23) `>`

EXPLANATIONS FOR EXAMPLE 5

(1) `DELETE 1-`

This command deletes all text and formatting requests that were entered as part of Example 4.

(2) OPTION TAB=;,5,30,34,36

The OPTION command is used to set tabulation stops at columns 5,30,34 and 36 and specify a tab control character. The tab control character is used to skip from one tab position to the next. The semicolon (;) is selected as the tab control character. You may specify any character that will not appear in the text which you will be entering.

Tabulation control with OS/32 TEXT is different from a typewriter. It is important that you remember these points:

- When you type the tab control character, the carriage (or cursor on a CRT) does not move to the actual column of a tab stop. All you will see is the tab control character you typed. OS/32 TEXT internally adjusts the text line to the selected tab stop. When you review your text using the TYPE command, the text appears in its tabular form.

- The second difference concerns the selection of a specific tab stop. Suppose you did enter text that extends beyond the first tab stop and you want to skip to the second stop to continue. On a typewriter, you would depress the tabulation key once. In OS/32 TEXT, you have to enter the tab control character twice. Once for the stop you passed, secondly for the stop where you want to continue typing!

- Each time you start OS/32 TEXT you must set the tabs if you wish to use tabs. OS/32 TEXT does not remember tab settings from a previous session.

(3) OPTION TAB

This command displays the currently specified tab control character and the tab stops.

(4) APPEND

This command tells OS/32 TEXT to accept and remember your text and formatting requests.

(5) .BLOCK BEGIN

This formatting request tells OS/32 TEXT where the in context block of text begins.

(6) .NOFILL

This formatting request allows the user to control the output format of text, in this case the format of the table. The request suspends filling lines, suppression of blanks, and right and left justification. Margin specifications and centering are adhered to.

It is important to remember that when .NOFILL is in effect, an input line cannot contain more characters than space is available in the output line. The availble space in the output line is determined by the line width specification minus the current margin settings.

(7) .CENTER

This formatting request causes the next line of text to be centered between the current margin settings.

(8) .LMARGIN +10

You adjust the left margin for the table.

(9) .INDENT -5

This formatting request causes outdenting of the next text line by five character positions. Outdenting positions the next line of text to the left of the current left margin setting.

(10) TO PRODUCE A;;400,000.00

This is a line of text for the table. Note that you have twice entered the tab control character ";", first to skip to position 5, then to position 30. Thus, "400..." will appear in the table at character position 30.

(11) NUMBER OF...;;;;bbbbb450.00

This is another text line of the table. Here you depressed the tab control character three times to skip over positions 5 and 30 to 34. At position 34 you type five spaces (b) and then the number 450.00. The spaces are required to compensate for the outdenting of this line by 5 character positions (see request on line 35).

(12) .BLOCK END

This formatting request delimits the end of the in context block.

(13) .LMARGIN RESET

You return the left margin setting to the value that was in effect prior to the beginning of the table.

(14) .FILL

This formatting request is the opposite of NOFILL. It restores the formatting features suspended in (7) by the NOFILL request.

FILL in conjunction with LJUSTIFY and RJUSTIFY, causes the output text lines to be left and right justified as you have seen throughout the examples.

(15) CR

You terminate input of text and formatting requests. OS/32 TEXT is now in command mode awaiting input of a command.

(16) TYPE 18-36

You review the text lines that were entered using the tab feature in OS/32 TEXT. The displayed lines now reflect the tabulation selected during typing. Note that the number column in line 36 is offset by the five spaces you entered. In the formatted output, the numbers will be aligned properly because this line is outdented by five character positions.

(17) SAVE EXAMPLE5.RV0

The SAVE command stores the text and formatting requests in a disc file named EXAMPLE5.RV0.

(18) FORMAT ,HALT

You request a formatted output of the text in example 5. Formatting will be halted at the beginning of each page until you depress RETURN to continue.

(19) This example...is blank!

This is the formatted output of page 1.

(20) CR

Continue with formatted output of next page.

(21) FACTORY COSTS...450.00

This is the formatted output of page 2 containing the in context table.

(22) If the table...preceding the table!

This is the formatted output of page 3.

(23) >

No more pages to output. OS/32 TEXT returns to command mode.

3.1.6 EXAMPLE 6: Merging Text, Automatic Dating and Request to "break" an Output Text Line

This example presents the use of three additional formatting features.

The first and most important one is the IMBED request. This request allows you to merge text from another disc file into the text of the document which you are currently formatting.

The second feature is a group of requests which perform automatic dating functions for you. They insert the date and time, which are maintained by the computer, into the formatted output of your document.

The third feature is the BREAK request. The BREAK request temporarily suspends filling of an output text line when FILL is in effect.

This example uses the text from Example 5 which was saved in file EXAMPLE5.RVO. The steps are:

- o Typing text and formatting requests
- o Imbedding text from disc file
- o Specifying BREAK in text
- o Requesting date and time

EXAMPLE 6

At the keyboard enter:

(1) >DELETE 1-

(2)
```
>APPEND
1  >.PLENGTH 23
2  >In this example we show how you can
3  >insert the text from a file into the formatted
4  >text output stream.
5  >We will use the text from example number 5
6  >which is contained in file EXAMPLE5.RVO
7  >and insert it after this sentence.
8  >.SKIP 1
```

```
(3)     9  >.IMBED EXAMPLE5.RV0

       10 >.SKIP
       11 >This passage will appear in the formatted output
       12 >following the text from the inserted file.
       13 >.SKIP 1
       14 >.LMARGIN +15
       15 >This example was concluded on

(4)    16 >.BREAK

(5)    17 >.MONTH

(6)    18 >.DAY ,

(7)    19 >.YEAR
       20 >at

(8)    21 >.TIME .
       22 >CR

(9)    >FORMAT ,HALT
       ENTER RETURN WHEN READY
       >CR
```

(10) In this example we show how you can insert the text from a file into the formatted text output stream. We will use the text from example number 5 which is contained in file EXAMPLE5.RV0 and insert it after this sentence.

(11) This example shows how to add a table as an "In Context Block" in your text. Since the table is too long to fit on the same page with this text, it is printed starting at the top of the next page.

Note that the remainder of this page is blank!

```
(12)  >CR

(11)                          FACTORY COSTS
          ADMINISTRATIVE COSTS
               TO PRODUCE                          400,000.00
               TO ENGAGE SECOND SHIFT              150,000.00
               MAINTENANCE COST PER MACHINE            250.00

          PRODUCTION COST
               PRODUCT A                               800.00
                    MATERIAL                           800.00
                    MACHINE HOURS                      200.00
                    MAN HOURS                          200.00

          LABOR COST
               FIRST SHIFT (PER MAN HOUR)                3.00
               SECOND SHIFT                              3.45

          NUMBER OF MACHINES AVAILABLE                 450.00

(12)  >CR

(11)  If the table had been specified as a "Floating Keep", this
      text would have been printed on the page preceding the table!

(13)  This passage will appear in the formatted output following
      the text from the inserted file.

(14)                This example was concluded on
(15)                AUGUST 24, 1978 at 13:28:26.

(16)  >
```

EXPLANATIONS FOR EXAMPLE 6

(1) DELETE 1-

This command deletes all text and formatting requests that were entered as part of Example 5.

(2) APPEND

This command tells OS/32 TEXT to accept and remember the text and formatting requests that you are about to enter.

(3) .IMBED EXAMPLE5.RV0

This formatting request specifies that the text to be formatted is contained in the file EXAMPLE5.RV0. The file contains text and formatting requests. The text from the file is inserted after line 8 of Example 6. When all text from the file has been formatted, processing continues with line 11 of Example 6.

(4) .BREAK

This formatting request is only effective if filling output lines is specified (.FILL). Here, a "break" occurs in the filling of lines at the end of the text on line 15 and a new output line is started.

(5) .MONTH

This request inserts the current month into the output text stream. The months are spelled out; e.g., JANUARY.

(6) .DAY ,

This request inserts the day of the month into the output text stream. The comma is optional and is added to the date for punctuation.

(7) .YEAR

This request inserts the year (e.g. 1978) into the output text stream.

(8) .TIME .

This request inserts the current time into the output text stream. The second period is optional and is added to the time display for punctuation.

(9) FORMAT ,HALT

You tell OS/32 TEXT to produce the formatted output of the text in Example 6. Formatting will be halted at the beginning of each page until you depress RETURN to continue.

(10) In this example...sentence

This is the formatted text from lines 2-6 that you entered in Example 6.

(11) This example...preceding the table!

This text was inserted from file EXAMPLE5.RV0.

(12) CR

Continue with formatted output of next page.

```
(13)  This passage...inserted file
```

This is the text from lines 11 and 12 that you entered in Example 6.

```
(14)  This example...on
```

This line is affected by the BREAK request. Note that a new line was output after the word "on".

```
(15)  AUGUST...13:28:26.
```

This line is the result of the date and time formatting requests.

```
(16)  >
```

No more pages to output. OS/32 TEXT returns to command mode.

CHAPTER 4
GUIDE TO USING OS/32 TEXT FEATURES

This chapter describes the functions that OS/32 TEXT performs. The commands and formatting requests used to produce a document are introduced according to their function in the preparation of a document. The commands and formatting requests are described in detail in Chapters 5 and 6, respectively.

The sections below discuss:

- terms which are frequently used throughout this manual
- functions performed by OS/32 TEXT commands
- preparing text to be formatted
- file handling

4.1 TERMINOLOGY

Throughout the manual the following words have special meanings:

- TEXT and OS/32 TEXT

  TEXT and OS/32 TEXT are used interchangeably to refer to the program whose operation and use is described in this manual.

- Command

  A command is a directive to the program to perform a certain function. For example, the words LOAD and START are commands to the Operating System. Whenever TEXT displays the > symbol, it expects you to enter a command to tell the program what to do. TYPE, APPEND, INSERT, FORMAT, etc., are OS/32 TEXT commands. Each OS/32 TEXT command is described in Chapter 5.

- Command Mode

  When TEXT expects you to enter a command, the prompt symbol (>) is displayed, and the program is said to be in the command mode. Some commands perform a function and then return the program to the command mode. GET, TYPE, and SAVE are such commands. Other commands switch the program to text mode (see below).

- Text Mode

  When you enter an APPEND, INSERT, or REPLACE command, OS/32 TEXT switches to text mode. In this mode, the program retains for future processing whatever is typed on the keyboard. In text mode lines of text and formatting requests can be entered. Text mode remains in effect until you switch back to command mode. This is done by depressing the RETURN key as the first entry on a new line.

- Formatting Request

  A formatting request is a directive which describes to TEXT the desired appearance of the document being produced. A formatting request is entered in text mode together with the document text. .MARGIN, .PARAGRAPH, and .SKIP are examples of formatting requests.

  A formatting request must always be entered on a line by itself and must appear at the beginning of that line. A formatting requests consists of a control character, a directive, and in some cases an argument, e.g., .SKIP 2, where the period is the control character.

  The control character which is initially in effect and is generally used, is the period. However, it is possible to redefine the control character with the .CONTROL formatting request. You may want to change the control character if you have a text line beginning with the current control character.

  Formatting requests are interpreted by OS/32 TEXT when the FORMAT command is entered. This command is used to produce formatted output (see below) of your document.

  Many formatting requests have initial value settings. These values are in effect each time the FORMAT command is executed. These values can be replaced by the values specified in the formatting requests appearing in the text.

  Chapter 6 describes the formatting requests and contains a table of formatting request initial values.

- Document Text and Raw Text

  These terms are used interchangeably to refer to the words, sentences, etc. which comprise the body of the document you are producing, exclusive of formatting requests.

- Text

  Text usually refers to both document text and formatting requests. In some cases the word text refers to raw text only. In these cases the meaning of text should be clear from the context.

- Formatted Output

  The term formatted output refers to the final form of the document you are preparing. The text which you have entered appears in the form you have specified using formatting requests. Formatted output can only be produced by entering the FORMAT command.

4.2 EXPLANATION OF COMMAND USAGE

The following sections describe the functions that OS/32 TEXT performs and the commands you would use to perform these functions. The commands are described in detail in Chapter 5.

4.2.1 Entering Text

The APPEND command is used to enter text to be processed. APPEND allows you to enter text when you are just starting to prepare a new document or to add text at the end of existing text.

When entering text, it is important that you remember the following:

- The last character on one line and the first character on the next line are treated as separate words. This means that you do not have to type a space to separate these words.

- For the same reason, do not hyphenate a word at the end of a line the way you would on a typewriter. Hyphenation requires special consideration. Please refer to the description of the .HYPHENATE formatting request in Chapter 6 for additional information.

4.2.2 Reviewing Text

After you have entered and modified text, you may want to review it. The TYPE command tells TEXT to display on the list device the text you have entered. If necessary, you can change the List device by using the OPTION command to specify a different list device (e.g., printer) for your review.

4.2.3 Modifying Text

To facilitate the correction of errors and text revision, a number of OS/32 TEXT commands are available:

- o FIND

  This command searches the text lines for a specified character string. Text lines containing a match are displayed.

- o CHANGE

  This command modifies text lines replacing a specified character string with a new one. The text is searched for a match of the specified character string. Each time a match is found, the line is modified and the resulting line is displayed.

- o INSERT

  This command allows you to insert text following a specified text line. The inserted text may consist of one or more lines.

- o INCLUDE

  This command is a powerful variation of the INSERT command. It allows you to copy one or more text lines and insert them following a specified text line. The lines to be copied may be part of:

  - the text currently being processed by OS/32 TEXT, or

  - the text stored in a disc file.

- o DELETE

  This command deletes one or more text lines.

4.2.4 Saving Text

The text that you entered can be saved in a disc file for future processing. The SAVE command creates a file using a file name that you specify and copies the text to this file.

OS/32 TEXT issues a warning message if you try to save text into an existing file. This is done to protect you from inadvertently destroying what may be a master file. It is possible to override this warning and save your text into an existing file.

4.2.5 Retrieving Text

Text that was stored in a disc file using the SAVE command can be retrieved using the GET command. The GET command copies the text from a specified file and makes it available to you as if you had just entered it from the keyboard.

4.2.6 Producing Formatted Text

To obtain the formatted output of the text which you have entered, you use the FORMAT command. This command produces the formatted output according to the formatting requests specified with the document text.

Before you can use the FORMAT command, the text to be formatted must be made available to OS/32 TEXT for formatting. This is done by either entering the text from the keyboard or using the GET command to retrieve the text that was previously entered and stored in a disc file.

The FORMAT command provides options to:

- o select the device where the formatted output should go (e.g., printer, Carousel, CRT, disc file).
- o select the formatted pages to be printed (e.g., page 1, pages 3-5, all pages).
- o halt printing at the beginning of each page to allow changing paper, until directed to continue.
- o output a cross reference listing that relates the text on the formatted pages to the line numbers of the text lines that you entered.

An important point to note is that the formatted output, if directed to a disc file, cannot be subsequently modified. The file can only be processed by the GET command and then the PRINT command to print the formatted document on a specified device.

4.2.7 Use of Cross Reference Line Numbers

The FORMAT command allows you to request a cross reference listing that relates the text on the formatted output pages to the raw input text lines of a document. The purpose of this listing is to aid you in making document text revisions.

The listing is generated following the formatted output of a document. It lists the line numbers for the first and last raw text lines that appear on each page. The line numbers correspond to the numbers assigned to these lines by TEXT during the editing process. They can therefore be used to easily locate the raw text lines for each formatted page. Note, however, that when the text being formatted contains fractional line numbers (e.g. 1.01, 5.07, etc.), it should first be SAVEd in a file and then retrieved with the GET command. This procedure will renumber the raw text lines and synchronize them with the line numbers in the cross reference listing.

4.2.8 Producing Copies of Formatted Text

If the formatted output of your text was directed to a disc file, you can use the PRINT command to make a copy of the file contents.

The PRINT command allows you to specify the output device where the file is to be printed.

Before you can use the PRINT command, the file to be processed must be made available to OS/32 TEXT via the GET command.

4.3 FORMATTING REQUEST USAGE

The following sections describe the formatting capabilities of OS/32 TEXT and the appropriate formatting requests to use when preparing a document.

4.3.1 Page Setup

The formatting requests in this group define the physical page size and text area for the formatted output.

4.3.1.1 Standard Page Layout

To obtain the formatted output of the document, the size and margins of the page that will contain the formatted document text must be defined.

OS/32 TEXT provides standard (default) page layout for paper size 8 1/2 inches by 11 inches. If this layout is not suitable for your needs, you can modify it by including the appropriate formatting requests in your text. Figure 4-1 shows the standard page layout used by OS/32 TEXT.

The outer rectangle of asterisks in the diagram outlines the physical dimensions of a sheet of paper. The inner rectangle of asterisks represents the text area: that part of the physical page into which OS/32 TEXT formats the document text.

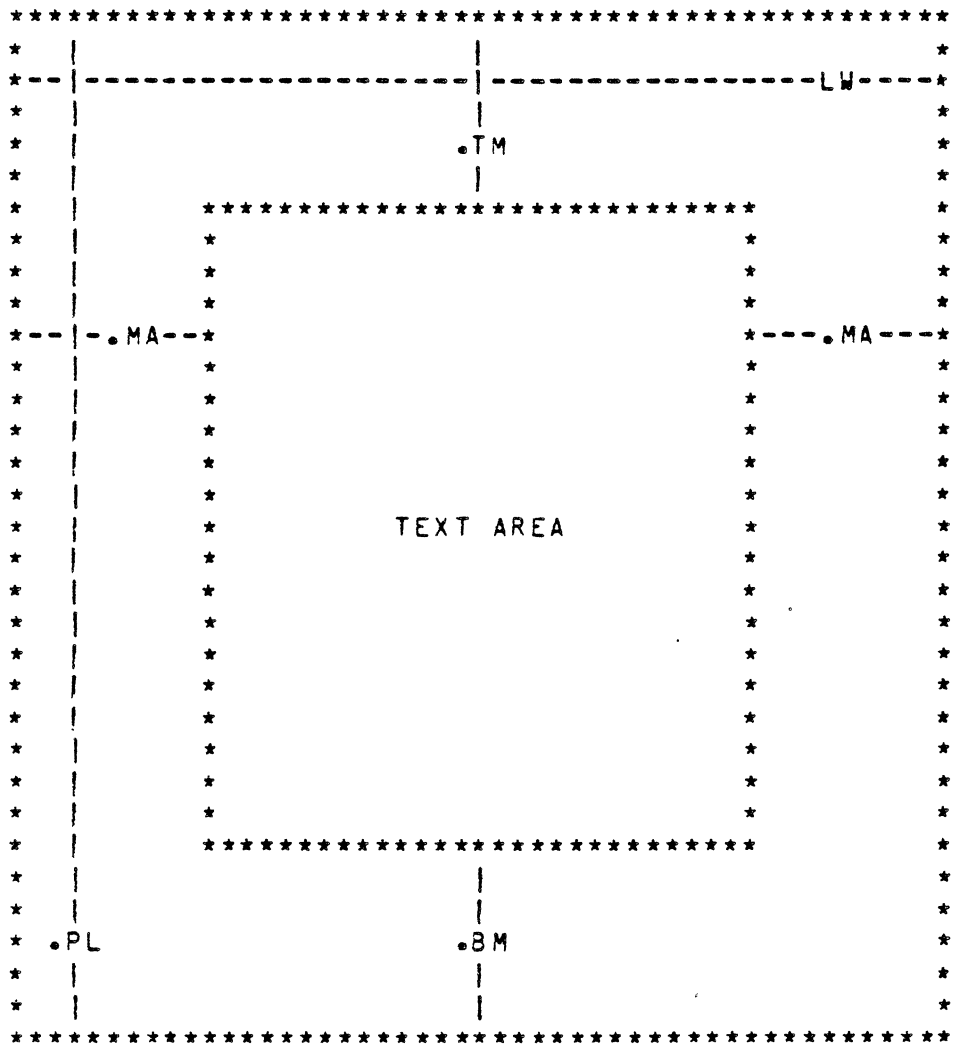

Figure 4-1 Standard Page Layout
(Paper Size 8-1/2 inches by 11 inches)

The Standard Page Layout has the following initial values for the formatting requests shown in the diagram:

.MA	12,74	(Left Margin = 12) (Right Margin = 74)	.LW	85	(Line Width)
.TM	5	(Top Margin)	.PL	66	(Page Length)
.BM	5	(Bottom Margin)			

4.3.1.2 Line Width

The LWIDTH formatting request specifies the physical width of the page that contains the formatted document. The line width is expressed in the number of character positions that fit across the page that you would be using.

4.3.1.3 Page Length

The .PLENGTH formatting request specifies the physical length of the page that will contain the formatted document. The page length is the number of lines available on one page of the output document.

4.3.1.4 Left and Right Margin

The .MARGIN formatting request sets the absolute left and right margins of a page. The margins are the available character positions on the physical page, counted from left to right.

The .PARAGRAPH and .INDENT formatting requests permit indenting and outdenting of the next line of text from the left margin.

4.3.1.5 Top and Bottom Margin

The .TMARGIN and .BMARGIN formatting requests define the number of blank lines to occur at the top and bottom of each page.

Top and bottom page titles, if specified, appear within these margins.

4.3.1.6 Left and Right Margin Temporary Offsets

The .LMARGIN and .RMARGIN formatting requests make adjustments to the left and right margin settings currently in effect. Each margin can be adjusted to the left or to the right.

The reset feature of these formatting requests returns the margins to the settings which were in effect prior to the current adjustment. Multiple adjustments/resets are permitted.

4.3.2 Text Line Processing

Formatting requests in this group control the conversion of raw text lines, which the user enters, to formatted output lines.

4.3.2.1 Line Filling

Unless the user specifies otherwise, the following formatting requests are in effect when TEXT is preparing formatted output:

- .FILL - fill output lines as specified by justification requests
- .LJUSTIFY ON - left justify output lines
- .RJUSTIFY ON - right justify output lines

These formatting requests cause words from the text that the user enters to be linked together into an output line until a word does not fit. (The word that does not fit becomes the first word of the next output line.) If necessary, the current line is then "justified" by inserting blank spaces between words so that the line touches both margins. When forming the output line, leading spaces and multiple spaces between words in the raw text are ignored. The placement of words into output lines may be optimized by defining word syllables in the raw text using the .HYPHENATE formatting request.

Certain formatting requests interrupt or "break" the stringing together of words. For example, a break occurs when starting a new paragraph. A break is also caused by a completely blank line in the raw text. A blank line appears in the formatted output (unless it occurs in a footnote). Formatting requests that cause a break are identified in Chapter 6 and in the formatting request summary in Appendix A.

The preceding paragraphs described the effect of .FILL with both .LJUSTIFY and .RJUSTIFY ON. The effect of .FILL with the latter two formatting requests in their various ON/OFF combinations is summarized as follows:

- .LJUSTIFY ON
- .RJUSTIFY OFF

  Lines of output text are aligned at the left margin. Words are placed in the line until a word does not fit, but no additional spaces are inserted between words.

- .LJUSTIFY OFF
- .RJUSTIFY OFF

  Text lines are output as if left justification only were in effect.

- .LJUSTIFY OFF
- .RJUSTIFY ON

  Output text is aligned at the right margin. No additional spaces are inserted between words.

- .LJUSTIFY ON
- .RJUSTIFY ON

  Lines of output text are aligned at both the left and right margins. Words are placed in the line until a word does not fit; additional spaces are inserted between words to cause margin alignment.

4.3.2.2 Sentence Terminators

The following punctuation characters are considered sentence terminators:

. ? ! :

These characters are only considered sentence terminators when they appear at the end of a word. If one of these characters appears in the middle of a word it is treated as part of the word itself.

Sentence terminators are always followed by two spaces in the formatted output lines. A brace and closing parentheses are also followed by two spaces if the character immediately preceding them is a sentence terminator.

All other punctuation characters are followed by one space only.

Punctuation characters and non-alphabetic characters are considered part of the word they appear with in the raw text, unless the characters are separated by blank spaces from the preceding word. In that case, they may become separated from the word and not even appear on the same output text line as the word. For example:

CHARACTERS

Part of a Word	Not Part of a Word
end.	end .
end,	end ,
(end)	(end)
"end"	" end "

The formatting request .BLANK can be used to override the implicit double spacing after a sentence terminator.

4.3.2.3 Centering Text

The .CENTER formatting request is used to center text lines between the left and right margin settings. Raw text lines are centered "as is" in the formatted output. Stringing together of lines and elimination of extra spaces between words is not performed.

4.3.2.4 Paragraphs

The .PARAGRAPH formatting request causes the start of a new paragraph in the formatted output text. .PARAGRAPH also performs optional in/outdenting of the first line in the paragraph.

4.3.2.5 Indenting

The .INDENT formatting request is used to indent or outdent a line relative to the current setting of the left margin.

4.3.2.6 Spacing and Skipping Lines

The .SPACE formatting request specifies the line spacing to be in effect in the formatted output. For example, .SPACE 2 specifies double spacing is to be in effect.

.SKIP outputs one or more blank lines directly following the .SKIP request.

4.3.3 Table Handling

This section provides information about building and formatting of document text in table form. It may be helpful to review Example 5 in Chapter 3 which addresses the subject of table handling.

4.3.3.1 Tabulation

OS/32 TEXT provides the capability to set tabulation stops which are recognized and adhered to during text entry and formatted output.

To produce formatted output of text in table form you specify the .NOFILL formatting request. This request suspends stringing words together in the output lines and does not remove multiple spaces between words in the raw text. It is your responsibility to arrange the table layout when the raw text for the table is entered.

The effect of .NOFILL with .LJUSTIFY and .RJUSTIFY ON or OFF on the formatted output is as follows: whenever left justification is in effect, lines of output text are positioned at the left margin; if only right justification is in effect, lines of output text are aligned at the right margin; if neither left nor right justification is in effect, lines are positioned at the left margin.

4.3.3.2 Placement

A table can be placed in two different ways in the formatted output of a document, using either the .BLOCK or the .KEEP formatting request. The .BLOCK request causes the table to be output where it appears in the raw text. When using the .KEEP request, text lines that follow the table in the raw text may appear before the table in the formatted output.

4.3.4 Page Headings, Footings, and Numbers

The formatting requests in this group control page labeling and numbering, and the processing of footnotes.

4.3.4.1 Top and Bottom Page Titles

The formatting requests .TTITLE and .BTITLE are used to define top and bottom page titles respectively, which are used during the formatting process. The specified titles appear on each output page in the areas defined as top and bottom page margins. If a page number symbol is part of a title, the output pages are automatically numbered.

4.3.4.2 Even-Odd Page Titles

The formatting requests .EVENTOP, .ODDTOP, .EVENBOTTOM, and .ODDBOTTOM, perform the same functions as .TTITLE and .BTITLE described above, except that they allow different titles to appear on odd and even numbered pages.

4.3.4.3 Automatic Page Numbers

The .PNUMBER formatting request is used to set the automatic page numbering. This request defines the initial page number, the page number increment and a special page number symbol. When this symbol appears in a page title, it is replaced by the computed page number in the formatted output of a document.

4.3.4.4 Footnotes

The text of a footnote is defined by the .FOOTNOTE formatting request. The text for the footnote should be entered immediately following the line that references it, so that it may appear at the bottom of the page containing the reference.

4.3.5 Document Revision Identification

When a document is revised, changes in the raw text can be identified using the .BAR formatting request in conjuction with the .RLEVEL formatting request. When producing the formatted output of revised document, revised areas are marked with revision bars.

The revision bars normally appear in the right margin unless ODD/EVEN page titles are specified. If ODD/EVEN page titles are specified, the bars appear in the right margin on odd numbered pages and in the left margin on even numbered pages.

The .BAR formatting request is used to define the modified text and revision level number when changes to a document are entered. For each subsequent revision of a document the user can increment the level number. The .RLEVEL formatting request specifies which revision level should be recognized so that the corresponding text is marked when producing the formatted output of the document. If .RLEVEL is not specified, the formatted output is not marked with bars.

4.3.6 User Control During Formatting

This section describes features that aid in maintaining text and producing the formatted output of a document.

4.3.6.1 RETURN

During processing of the FORMAT command, you can direct TEXT to temporarily read text that you will enter at the command input device. The RETURN formatting request is used to do this. The text that you enter becomes part of the formatted output. This feature may be used, for example, when preparing form letters, to insert a different address in each letter.

Optionally, .RETURN can display a comment on the command input device to prompt you on the type of text to enter.

When .RETURN is used, the device for the formatted output cannot be the same as the command input device. If you have no alternatives, direct the formatted output to a disc file and use the PRINT command to obtain copies of the formatted output.

4.3.6.2 QUIT

When OS/32 TEXT is creating the formatted output of a document, the formatting process may be terminated via the .QUIT formatting request. The effect of .QUIT depends on where it is encountered during the formatting process. The formatting requests .IMBED, and .RETURN, and the command FORMAT are affected by .QUIT.

4.3.6.3 COMMENT

The .COMMENT request allows you to insert comments throughout the raw text of a document to keep notes for your own use concerning the preparation, maintenance, and formatting requirements of a particular document.

The .COMMENT formatting request has no effect on the appearance of the formatted document. When producing the formatted output, it is ignored.

4.3.6.4 Prompt Suppression

MTM displays a dash "-" prompt when OS/32 TEXT performs Input/Output operations to an MTM terminal (CRT, Carousel, etc.). If you are using the terminal (e.g. Carousel) to produce the final formatted versions of a document, the display of this prompt is generally undesirable.

The following OS/32 MTM commands allow you to control the display of the prompt:

```
ENABLE PROMPT
PREVENT PROMPT
```

Since these are MTM system commands, they can only be issued in the MTM System Command Input Mode which is indicated by the display of a "*".

Normally, you would enter these commands before you load or start OS/32 TEXT. If you have already started OS/32 TEXT you can use the following procedure to enter the ENABLE or PREVENT commands.

Assume that you want to disable the prompt: (OS/32 TEXT must be in the command input mode.)

USER/MTM DIALOGUE	EXPLANATION
>PAUSE	Pause OS/32 TEXT to gain MTM command mode
*TASK PAUSED	OS/32 TEXT is paused and MTM awaits command
*PREVENT PROMPT	Disable display of -
*CONTINUE	Continue operation of OS/32 TEXT
>	OS/32 TEXT awaits command; note that the - is no longer displayed.

If executing TEXT at the system console, the task identification displayed on the system console device cannot be suppressed.

It is therefore not practical to use the console device to produce a formatted output of a document. The output should be directed to a printer or another terminal on the system.

4.3.6.5 Changing Paper

The HALT operand of the FORMAT command provides the capability to suspend output of formatted text at the beginning of each page.

This capability allows you to change or position the paper in a printing device if the paper has to be inserted manually for each page. To resume producing formatted output after a halt, depress the RETURN key.

Please note that if the formatted output is directed to a disc file and then printed via the PRINT command, printing cannot be suspended after each page.

4.3.7 Use of the Imbed Formatting Request

The .IMBED formatting request is a powerful feature of OS/32 TEXT. When producing the formatted output of a document, .IMBED tells TEXT to switch to a different file to obtain text to be formatted. The text in the file specified in the .IMBED request is formatted and merged into the document being formatted. The file being switched to may itself contain an .IMBED request for yet another file, and so on up to a level of ten files.

Formatting an imbedded file is terminated when an end-of-file or the .QUIT formatting request is encountered. Formatting continues, using the text that contained the .IMBED request.

Normally, the .IMBED request specifies a disc file as the input file. However, the command input device can be specified. The information contained in a file to be imbedded may consist of raw text only, formatting requests only, or a combination of these two.

4.3.7.1 Merging Text

Merging text from different files into a single document is one of the most common applications of the .IMBED formatting request. When producing a manual such as this one, each chapter or section can be maintained in a separate file. When it is necessary to produce the formatted output of the entire document, the chapters are combined into a single document by building a file which contains .IMBED requests for each individual file.

4.3.7.2 Predefined Sets of Formatting Requests

Another application of .IMBED is to specify a file which consists completely of formatting requests. For example, you could define a standard page layout, spacing and title for a particular document (e.g., company memo), and save the formatting requests in a file. When you want to produce a formatted output of the document, you simply imbed this file instead of entering all the formatting requests each time with the document text.

Of course there are many more applications besides the ones mentioned here. In general, if you find yourself typing a series of formatting requests or text (e.g., copyright statement, distribution list) repeatedly, store these items in a file and insert them as necessary using the .IMBED formatting request.

4.4 FILE HANDLING

A file is a collection of data stored on a disc device. The stored data can be retrieved whenever it is needed for processing. Files remain on the disc until they are explicitly removed.

The following aspects of file handling are considered when using TEXT:

- naming a file in which to store text;
- specifying a file from which to retrieve text;
- displaying names of existing files;
- deleting obsolete files.

4.4.1 Naming a File

Naming a file to be used to store text is done when using the SAVE and FORMAT commands. These commands create a file from a specified name and output text to that file.

Chapter 5 explains the format of the file descriptor operand (fd) which is used in these commands to specify a file name.

4.4.2 Specifying a File Name to Retrieve Text

The GET command is used to retrieve text from a file previously specified in the SAVE or FORMAT command. The rules for specifying a file name are the same as for naming a file and are described in Chapter 5.

4.4.3 Displaying File Names

At times, you may want to review the various files which you have created using TEXT.

The Operating System command DISPLAY FILE is used to display the names of existing files. To use the command, the System must be in command input mode, as indicated by the display of an asterisk "*". Normally, you would use the command before starting or after terminating the execution of TEXT.

However, it is possible to temporarily halt the execution of TEXT using the PAUSE command to switch control to the Operating System command input mode. You can then review your files. For additional information refer to the MTM Terminal User's Reference Manual or the OS/32 MT Operator's Reference Manual.

4.4.4 Deleting Files

The Operating System command DELETE is used to remove files that are no longer needed.

The general format of the command is:

```
DELETE  fd
```

where:

```
fd = the file descriptor identifying
     the file to be deleted.
```

The use of fd is described in Chapter 5. For additional information refer to the MTM Terminal User's Reference Manual or the OS/32 MT Operator's Reference Manual.

CHAPTER 5
COMMAND DESCRIPTIONS

5.1 INTRODUCTION

This section describes the use of OS/32 TEXT commands. The general usage of OS/32 TEXT commands is explained in Chapter 4.

Section 5.2 describes the command syntax. Section 5.3 contains descriptions of each command. The descriptions are presented in alphabetical order.

5.2 SYNTAX

This section describes the syntax used to specify a TEXT command.

5.2.1 Command Format

The general format of a TEXT command is as follows:

MNEMONIC operand1,operand2...

where:

MNEMONIC specifies a TEXT command. The underlined part of the mnemonic is the minimum abbreviation acceptable for that command. The mnemonic must be separated from its first operand by at least one blank.

The underlined portion of an operand refers to the minimum abbreviation of a keyword operand. Operands are separated by commas.

TEXT commands, and operands which do not explicitly reference text strings, may be entered in upper or lower case letters.

5.2.2 Operand Formats

This section describes the various forms of operands that may be specified to reference text lines. The operand symbols given in the following sections are used in the command descriptions to indicate which operands may be used with a specific command.

5.2.2.1 String Operand

Symbol: S

A string operand specifies a text line by a sequence of one or more ASCII characters - including blanks.

The string operand may be one of the following forms:

- delimiter S delimiter
- delimiter S delimiter column

where:

"delimiter" is a non-alphanumeric character except a comma, minus sign, or blank. The delimiter may not be part of the string it defines.

"column" is an integer specifying that the string is only valid for that character position in a text line. Matching strings that do not start in the specified column are ignored.

EXAMPLES OF STRING OPERANDS:

1. .EXAMPLE.

   specifies a string of seven characters. The period is used as the delimiter.

2. /used/4

   specifies a string of four characters. The string is defined only for column 4 of the line of text being searched.

5.2.2.2 Line Number

Symbol: Line Number

A line number operand specifies a text line by its assigned number. The specified line number may be one of the following forms:

a) integer

b) integer.decimal

where:

$0 \leq integer \leq 99999$
$0 \leq decimal \leq 99$

EXAMPLES OF LINE NUMBER OPERANDS:

1. Valid line numbers:

```
   32
    1
99999
    7.99
  426.00
 1400.7
    0.23
```

2. Invalid line numbers:

```
     6.839    too many decimal places
100000        integer too large; maximum is 99999
     2.       missing decimal digit
  -615        negative number, illegal
    .4        missing integer part
```

5.2.2.3 L Operand

Symbol: L

The L operand specifies that text lines may be referenced either by a string or by a line number as described above.

5.2.2.4 Range Operand

Symbol: R

The range operand specifies a range of text lines (i.e., from-to) which are to be processed by the command containing the range operand.

The lines referenced by R are specified by the L operand, which is either a line number or a string operand (S). The R operand may have the following forms:

R*	Description
L1	Line1 only
L1-L2	Line1 thru Line2 inclusive
L1-	Line1 thru last line of text
-L2	Current line thru Line2.
(blank)	Omitting the Range operands implies current line only except for the SAVE command, where, if no Range is given, all lines of text are processed.
-	Current line through last line of text.

*The dash (-) is used as a range argument separator.

EXAMPLES OF RANGE OPERANDS:

Range	Type	Description
1-20	L1-L2	Lines numbered 1 through line 20 including all fractional (e.g. 1.01,2.10) lines in between
.LA.-.END.	L1-L2	Line containing string

		LA through line containing string END, inclusive
8.3-.RETURN.	L1-L2	Line numbered 8.3 through line containing RETURN inclusive
73	L1	Only line numbered 73
.as is.10	L1	Only line containing string "as is" beginning in column 10

5.2.2.5 File Descriptor Operand

Symbol: fd

The operand symbol fd refers to a file descriptor in standard OS/32 and OS/32 MTM format. The fd may specify a device or a file on a mass storage device.

The general format of fd has the following parameters:

voln:filename.ext/efd

voln:

The volume name is composed of one to four alphanumeric characters, terminated by a colon. The first character must be alphabetic. This is the name of the volume on which the file resides. If the volume name is omitted, the default volume name is assumed.

The default volume can be set with the VOLUME system command. For a description of this command, see the appropriate reference manual for your system as listed in the Preface of this manual.

When fd refers to a device, the VOLN field is used to contain the device mnemonic. The device mnemonic can be from one to four characters long and is terminated by a colon. A filename and extension, if entered, are ignored.

filename

The filename consists of one to eight alphanumeric characters where the first character must be alphabetic. This is the identifier for the file and may be anything the user choses.

.ext

The extension consists of a period followed by one to three alphanumeric characters. The extension may be omitted entirely, or just the period given, in which case the extension is considered to consist of blanks. The extension can be used to denote the contents of the file such as type of data, revision level, etc.

/efd

The optional extended file descriptor is recognized by MTM only. The efd can be P, G, or S, which stand for Private, Group, or System file respectively. If the extended field is not given, the default is P, for Private.

EXAMPLES OF FD OPERANDS

fd	voln:	filename	.ext	/efd
M300:FILE.RV0	M300:	FILE	.RV0	Private file by default
EXAMPLE.1/G	default volume	EXAMPLE	.1	Group File
STDMEMO	default volume	STDMEMO	(blank) by default (b=blank)	Private file by default
M67A:F1234567/G	M67A:	F1234567	(blank)	Group File
PR:	device:printer	N/A	N/A	N/A
CON:	device:console	N/A	N/A	N/A

5.3 DESCRIPTION OF TEXT COMMANDS

This section describes OS/32 TEXT commands. The commands are presented in alphabetical order.

With the exception of FORMAT and PRINT, these commands are based on the capability provided by the OS/32 EDIT program. You may refer to the OS/32 User's Guide to supplement the information provided in this section.

Most command descriptions are illustrated with examples. It is implied that the commands and text that the user inputs in these examples are always terminated by depressing the RETURN key. In the examples CR is used when the user should depress RETURN as the only entry on the given line.

The command descriptions also refer to the TEXT concepts of "Line Numbers" and "Current Line" which are defined here.

- Line Numbers

  Line numbers are a convenient mechanism for identifying text lines. The line numbers can be used by most TEXT commands to reference text for editing functions.

  The line numbers are automatically assigned when a file is opened for editing, or are generated when text is entered from the command input device. Each line retains its assigned number as long as the line exists during the text processing session. Line numbers are output only when text is displayed. They are not written to the output file.

  Line numbers are assigned starting with number 1 and are incremented by one up to a maximum of 99999.

  Fractional line numbers are assigned when text is inserted between integer line numbers. They are generated by incrementing the line number, which precedes the inserted text, by .01. The line number of each inserted line is .01 greater than the previous line. Incrementing stops if a new line number would equal an existing line number. Thus, several lines may exist with the same line number. Text may be renumbered by saving the text to a file and then retrieving it again (see SAVE and GET commands).

- Current Line

  TEXT maintains a "current line" during processing. The current line is defined as the result of performing a TEXT command. Most TEXT commands can perform their operation on the current line as the default case. A command may also specify any line which is not the current line and thus change the current line. The command descriptions give the

status of the current line at the end of command execution.

5.3.1 APPEND Command

Mnemonic: APPEND

Operands: None

Description: Allows user to enter text from the command input device. Line numbers and a ">" symbol are output as prompts to the user. Depressing RETURN directly after the prompt is given terminates text entry and returns TEXT to command mode. If no text currently exists, the new text starts with line number one (1). If text currently exists, new lines are added to the end of the existing text.

Current Line: Last line of text entered.

EXAMPLES:

Example 1:

No Existing Text:

```
Enter command:  >A

TEXT prompts:   1 >XYZ                          User enters:
                                                XYZ
TEXT prompts:   2 >ABC                          User enters:
                                                ABC
TEXT prompts:   3 >NO MORE                      User enters:
                                                NO MORE
TEXT prompts:   4 >CR                           User enters:
                                                Carriage Return

Resulting text:     1     XYZ
                    2     ABC
                    3     NO MORE               Current Line
```

Example 2:

```
Existing text:      1     FILE                  Current Line
                    2     IS
                    2.30  ASCII TYPE

Enter Command:  >A

TEXT prompts:   3 >TODAY                        User enters:
```

```
                                                    TODAY
                    4 >CR                           User enters:
                                                    Carriage
                                                    Return

Resulting text:     1     FILE
                    2     IS
                    2.30  ASCII TYPE
                    3     TODAY                     Current Line
```

5.3.2 CHANGE Command

Mnemonic: CHANGE

Operands: S1,S2,R

S1=String1
S2=String2
R=Range

Description: This command replaces every occurrence of string S1 with string S2 in the range R.

S1 specifies the string to be replaced while S2 defines the new string. S2 may specify an empty string. All changed text lines are output to the list device so that the user can verify the change. See OPTION VERIFY.

Current Line: Last Line of Range.

EXAMPLES:

Example 1:

Change "was" to "is" in line 1.

```
Existing text:      1    *This was an example      Current Line
                    2    *how to use the the
                    3    *CHANGE command.

Enter command:  >CH/was/,/is/1

Resulting text:     1    *This is an example       Current Line
                    2    *how to use the the
                    3    *CHANGE command.
```

Example 2:

Delete superfluous "the" in line 2
starting at character position 17.

```
Enter command:  >CH.the.17,//,2

Resulting text:     1    *This is an example
                    2    *how to use the           Current Line
                    3    *CHANGE command.
```

Example 3:

Change "*" to "!" in all text lines.

Enter command: >CH/*/,.!.,1-

Resulting text:
```
    1    !This is an example
    2    !how to use the
    3    !CHANGE command.          Current Line
```

5.3.3 Change COLUMN Command

Mnemonic: COLUMN

Operands: C,S,R

C=Column
S=String
R=Range

Description: The current contents of each line in the range R are overwritten beginning at the column specified by C, with the string specified by S. When the COLUMN command is executed, changed text lines are output to the list device so that the user can verify the change. See OPTION VERIFY.

Current Line: Last line of Range.

EXAMPLES:

Example 1:

Change column 9 in line 1.

```
Existing text:     1    ABCDEFG 11234          Current Line
                   2    ABCDEFG 56789

Enter command:  >CO 9,/0/,1

Resulting text:    1    ABCDEFG 01234          Current Line
                   2    ABCDEFG 56789
```

Example 2:

Change columns 1-3 in all lines.

```
Enter command:  >CO 1,/XYZ/,1-

Resulting text:    1    XYZEFG  01234
                   2    XYZEFG  56789          Current Line
```

5.3.4 DELETE Command

Mnemonic: DELETE

Operands: R R=Range

Description: This command deletes the text lines specified by the range "R".

Current Line: Line following last line which was deleted. If last line of the file is deleted, then the last existing line of text becomes the current line. The current line is undefined if all text is deleted.

EXAMPLES:

Example 1:

Delete line 1.

```
Existing text:      1    Text Line AA          Current Line
                    2    Text Line BB
                    3    Text Line CC

Enter command:  >DE 1

Resulting text:     2    Text Line BB          Current Line
                    3    Text Line CC
```

Example 2:

Delete all lines.

```
Enter command:  >DE 2-

Resulting text:     none                       Current Line
                                               undefined
```

5.3.5 END Command

Mnemonic: END

Operands: none

Description: Allows user to return to operating system control. The editor goes to "end of task" with a return code of 0 to signify normal termination.

If the user issues END before storing any current text on a file (using the SAVE command), a reminder message is output.

5.3.6 FIND Command

Mnemonic: FIND

Operands: S,R — S=String, R=Range

Description: Searches for all occurrences of the string specified by S in the Range R, and outputs to the list device the lines containing S.

Current Line: Last line of Range

EXAMPLES:

Example 1:

Find line containing string "AA".

```
Existing text:      5    *SEARCH FOR          Current Line
                    6    $SYMBOL
                    7    *AA
                    8    *BB
                    9    $CC

Enter command:  >F/AA/,5

                STRING NOT FOUND             Information
                                             message
                                             Line 5 does
                                             not contain
                                             "AA".
                                             Current Line
                                             is still 5.

Enter command:  >F/AA/,5-9

List Device Output:  7  *AA                  TEXT found line
                                             Current Line
                                             is 9
```

Example 2:

Find all lines starting with a "*" in column 1.

```
Enter command:  >F/*/1,5-

List Device       5  *SEARCH FOR                TEXT found lines
Output:           7  *AA                        Current Line
                                                is 9
                  8  *BB
```

5.3.7 FORMAT Command

Mnemonic: FORMAT

Operands: [fd] { [,P] / [,P-P] } [,HALT][,XREF]

fd=file descriptor
P=page number(s)
HALT=Enable end-of-page stop
XREF=Enable cross ref. listing

Description: This command is used to produce the formatted output of a document. The appearance of the output is based on the document text and the associated formatting requests. Before entering the FORMAT command the text to be formatted must be available to OS/32 TEXT. That is, it must have been entered from the command input device (e.g., CRT) or retrieved from a file using the GET command.

Each time the FORMAT command is issued, the initial values for certain formatting requests are set up (see Chapter 6).

Formatting begins with the first line of the input text (consisting of document text and formatting requests) and continues until the last input text line has been processed or the .QUIT (see Chapter 6) formatting request is encountered.

If the formatted output of a document was directed to a file, that file cannot be used as input to OS/32 TEXT. The OS/32 TEXT PRINT command is used to copy or print a file containing formatted text.

All operands except fd may be entered in any order.

The brackets enclosing the command operands indicate that the use of each operand is optional.

fd

The operand fd specifies the file name or device to which the formatted text is output. When fd is omitted, formatted output is directed to the current list device.

When fd specifies a file, the following considerations apply:

- o if fd does not exist, an indexed file (of record length 132 bytes) is allocated for the formatted output.
- o if fd exists, the actions taken by OS/32 TEXT depend on its current operation mode.
  - interactive mode

    if the specified fd was used in the last GET command, the FORMAT command is rejected. Otherwise, a warning message is output. The user then has the option to override the warning or to enter a new fd.

  - batch mode

    if the specified fd was used in the last GET command, the FORMAT command is rejected and program execution aborted. Otherwise, the file specified in fd is deleted and an information message is output. The specified file is then allocated as described above.
- o the file is closed upon completion of the FORMAT command.

P or P-P

The braces indicate a choice of alternate operands. The operand P specifies a single page number. The operand "P-P" specifies the beginning and ending page of a group of consecutive pages. When the operand is omitted, all pages are output.

HALT

This option is useful when paper must be fed manually to a printing device, or when reviewing formatted text before printing final copies. The operand "HALT" specifies that output of formatted text is halted temporarily

at the beginning of each page. Prior to output of the first page, the message "ENTER RETURN WHEN READY" is displayed as a reminder that the HALT option was selected. This option is effective only if the formatted output is directed to an interactive device. To resume formatted output after a halt, the RETURN key must be depressed.

XREF

The operand XREF causes OS/32 TEXT to generate a cross-reference listing at the end of the formatted output. This listing relates the text on the formatted output pages to the line numbers assigned by OS/32 TEXT to the raw text and formatting requests entered by the user.

This option aids the user in locating text passages, etc. in the raw text to facilitate document revisions.

EXAMPLES:

1. Output entire document to current list device.

   Enter command: >FO

2. Output page 2 of document to printer.

   Enter command: >FO PR:,2

3. Output pages 2-10 of document to current list device and halt before outputting each page.

   Enter command: >FO ,2-10,H

4. Output entire document to FILE.A on default volume and produce a cross-reference listing. FILE.A does not exist.

   Enter command: >FO FILE.A,X

5.3.8 GET Command

Mnemonic: GET

Operands: fd — fd=file descriptor

Description: This command deletes existing lines, then opens the specified file and makes the file contents available for processing. The file itself is not changed during processing. If the GET command is issued before the current text is saved, a reminder message is output to the user.

Current Line: First line of text.

EXAMPLES:

Example 1:

Get a file from default volume.

Enter command: >G EXAMPLE.1

Resulting text: Text from file EXAMPLE.1 is available for processing. — Current Line is 1

Example 2:

Get a file from volume M300.

Enter command: >G M300:EXAMPLE.2

Resulting text: Text from file EXAMPLE.2 is available for processing. — Current Line is 1

5.3.9 INCLUDE Command

Mnemonic: INCLUDE

Operands:	L,fd,R	L=Line operand
	,fd,R	fd=file name
	L,,R	R=Range
	,,R	

Description: This command inserts text lines in the text currently available. The lines to be inserted may come from two sources:

1. a file

2. from within the current text.

Include lines from a file:

Lines specified by R from file fd are inserted after line L. If L is omitted, the inserted lines follow the current line.

Include lines from within the current text:

When fd is omitted, the lines specified by R are taken from the current text. The lines specified by R are inserted after line L. If L is omitted, the inserted lines follow the current line.

The original lines specified by R are still available.

Current Line: Last line included.

EXAMPLES:

Example 1:

Include all lines from file EXAMPLE.1 after line 2.

```
Existing Text:      1    BEGIN
                    2    THIS IS AN
                    3    END OF
                    4    EXAMPLE
                    5    NUMBER 1.
                    6    *                    Current Line

Enter Command:  >INC 2,EXAMPLE.1,1-
```

```
Resulting text:     1     BEGIN
                    2     THIS IS AN
                    2.01  EXAMPLE HOW TO
                    2.02  USE THE INCLUDE
                    2.03  COMMAND.
                    2.04  *                          Current Line
                    3     END OF
                    4     EXAMPLE
                    5     NUMBER 1.
                    6     *
```

Example 2:

Include lines 4,5,6 from current text after line 1.

```
Enter command:  >INC 1,,4-6
```

```
Resulting text:     1     BEGIN
                    1.01  EXAMPLE
                    1.02  NUMBER 1.
                    1.03  *                          Current Line
                    2     THIS IS AN
                    2.01  EXAMPLE HOW TO
                    2.02  USE THE INCLUDE
                    2.03  COMMAND.
                    2.04  *
                    3     END OF
                    4     EXAMPLE
                    5     NUMBER 1.
                    6     *
```

5.3.10 INSERT Command

Mnemonic: INSERT

Operands:	L	L=Line
	none	operand

Description: This command allows text lines to be entered from the command input device. The new text is inserted after the line specified by the L operand. If L is omitted, the text is inserted after the current line. Line numbers followed by the > symbol are output as prompts to the user. Lines can be inserted until text input is terminated by depressing RETURN directly after the prompt is given.

Current Line: Last line inserted.

EXAMPLES:

Example 1:

Insert text after the Current Line.

```
Existing text:      1    AAA
                    2    DDD
                    3    EEE                        Current Line

Enter command:  >INS

TEXT prompts:   3.01 >FF                            User enters:
                                                    FF
TEXT prompts:   3.02 >GG                            User enters:
                                                    GG
TEXT prompts:   3.03 >CR                            User enters:
                                                    Carriage Return

Resulting Text:     1    AAA
                    2    DDD
                    3    EEE
                    3.01 FF
                    3.02 GG                         Current Line
```

Example 2:

Insert text after line 1.

Enter command: >INS 1

TEXT prompts:	1.01 >B	User enters: B
TEXT prompts:	1.02 >C	User enters: C
TEXT prompts:	1.03 >CR	User enters: Carriage Return

Resulting text:

```
1     AAA
1.01  B
1.02  C                    Current Line
2     DDD
3     EEE
3.01  FF
3.02  GG
```

5.3.11 OPTION Command

Mnemonic: OPTION

Operands:		See Section
	.LIST=fd	5.3.11.1
	.NO_TAB	5.3.11.2
	.NOVERIFY	5.3.11.3
	.TAB=character, stop1,stop2.....	5.3.11.4
	VERIFY	5.3.11.5

Current Line: Unchanged

Description: Only those operands which pertain to the use of TEXT are discussed below. You may refer to the OS/32 EDIT User's Manual for information about the operands which are not discussed.

As many operands as desired can be entered and in any order. Operands must be separated by commas.

OS/32 TEXT processes the options from left to right. Therefore, if conflicting operands are entered in the same option command, the last operand given is in effect.

When OS/32 TEXT is first executed, the options are set to the following default values:

```
BLOCK = 5
LENGTH = 80
LIST = determined by START arguments
MODE = ASCII
NOLOG
NOTAB
TERMINATOR = unspecified
VERIFY
```

The values of all current options may be displayed by entering the OPTION command with no operand.

The current settings of the operands BLOCK, LENGTH, LIST, LOG, MODE, TERMINATOR, and TAB may be individually displayed by entering OPTION followed by the desired operand only, e.g. O LIST.

EXAMPLES:

Example 1:

Request display of all option settings after starting OS/32 TEXT.

Assume List Device is CON:

```
Enter Command:  >O

TEXT outputs:       MODE=ASCII
                    LENGTH=80
                    LIST=CON:
                    BLOCK=5
                    TERMINATOR=UNSPECIFIED
                    NOLOG
                    VERIFY
                    NO TABS
```

Example 2:

Change three options and display all option settings.

```
Enter command:  >O NOV,LE=120,TA=$,10,16

Enter command:  >O

TEXT outputs:       MODE=ASCII
                    LENGTH=120
                    LIST=CON:
                    BLOCK=5
                    TERMINATOR=UNSPECIFIED
                    NOLOG
                    NOVERIFY
                    THE TAB CHARACTER IS $
                    10 16
```

Example 3:

Display current option setting for MODE.

```
Enter command:  >O MO

TEXT outputs:       MODE=ASCII
```

5.3.11.1 Operand:LIST=fd fd=file descriptor

Description: Causes the LIST device to be assigned to fd. The responses generated by TEXT commands are output to fd. At times it may be useful to direct the display produced by TYPE to a line printer in order to obtain a hard copy of the current text.

EXAMPLE:

Start OS/32

TEXT:	START, LIST=PR:, COMMAND=CON:	All list output is directed to PR:
TEXT starts:	OS/32 TEXT Rn	n=revision number
Enter Command:	>0 LI=CON:	All list output is directed to CON:

5.3.11.2 Operand: NOTAB

Description: Current Tab settings are cleared. Text already entered using tab control is not affected.

EXAMPLE:

Enter command:	>0 NOT	Clear tab settings
Enter command:	>0 TA	Display tab settings
TEXT outputs:	NO TABS	No tabs in effect

5.3.11.3 Operand: NOVERIFY

Description: This operand cancels the VERIFY option. "changed" lines are not output during execution of the CHANGE or COLUMN command.

5.3.11.4 Operand: TAB
TAB=Char,T1,T2...

Char=Tab Character
Tn=Tab Stop

Description: The TAB operand allows the user to enter text in columnar form. The tab character is used to indicate where tabulation should occur. The text is expanded by OS/32 TEXT before it is stored for subsequent use. When text is saved or output to the terminal, it is always in expanded form.

TAB

Without arguments displays the current Tab character and settings, or "NO TABS".

TAB=Char,T1,T1...

Defines the Tab Character "char" and up to 20 tabulation stops, "T1,T2" etc. The Tab Character allows you to skip from one tab stop to the next. "Char" should be a character that does not occur in your text. The reverse slash "\" or "CTRL I" (depress these keys simultaneously) are recommended. If "CTRL I" is selected, the "TAB" key may be used to skip to the tab stops. The tab stops "T1,T2..." may be entered in any order. OS/32 TEXT will arrange them in numerically ascending order. Tab stops may be added to existing ones without redefinition of established stops. However, the Tab Character must be entered every time a Tab Stop is entered!

EXAMPLES:

Example 1:

Define tabs and enter text using tabs.

```
Enter command:  O TA=\,20,10                    Define Tab
                                                Character and
                                                set two TAB
                                                STOPS.

Enter command:  >O TA                           Request
                                                display
                                                of tabs

Text outputs:   THE TAB CHARACTER IS \
                     10 20

Enter Command:  >A                              Request text
                                                input mode

TEXT prompts:   1>DATE\DEPOSIT\NEW SUM          Enter text
                                                using
                                                tab stops

TEXT prompts:   2>CR                            Enter "CR"
                                                (carriage
                                                return)
                                                to end
                                                text input.

Enter command:  >T1                             Display
                                                line 1.

TEXT outputs:       1    DATE      DEPOSIT    NEW SUM
```

Example 2:

Add a tab stop to existing tabs.

```
Enter command:  >O TA=\,15

Enter command:  >O TA

TEXT outputs:       THE TAB CHARACTER IS \
                     10 15 20
```

5.3.11.5 Operand: VERIFY

Description: A line which is modified during execution of the CHANGE or COLUMN command is output to the terminal, when VERIFY is in effect.

5.3.12 PAUSE Command

Mnemonic: PAUSE

Operands: None

Description: Allows user to return to operating system control, temporarily. To return to TEXT, the user issues the CONTINUE command. Upon re-entering TEXT, all text is available as before the PAUSE command was given.

Current Line: Unchanged

EXAMPLE:

Enter command: P

System outputs: TASK PAUSED

Enter command: CO

The OS CONTINUE command

5.3.13 PRINT Command

Mnemonic: PRINT

Operands: fd — fd=file descriptor

none

Description: This command is used to print a file that was previously produced using the FORMAT command. The PRINT command can only print a file that contains formatted text. The operand fd specifies the device to which the text is sent. Before a formatted file can be printed, it must be made available to OS/32 TEXT via the GET command.

A formatted file can only be printed or copied to another file; it cannot be edited. The PRINT command is used for printing and copying. The PRINT command provided by TEXT should not be confused with the MTM PRINT command.

fd

The operand "fd" specifies the device or file name to which the previously formatted text is to be output.

Normally, a device is specified to obtain a hard copy of the document. The device must be able to handle the line width which was selected to create the formatted output. Otherwise, loss of data will occur. The PRINT command does not perform any formatting to fit the text lines to different devices.

When fd specifies a file, the resulting output is a copy of the input file. Processing of the file by OS/32 TEXT is subject to the following considerations:

a) if fd does not exist, an indexed file of record length 132 bytes is allocated for the formatted output.

b) if fd exists, the actions taken by OS/32 TEXT depend on its current operation mode.

ba) interactive mode

if the specified fd was used in the last GET command, the PRINT command is rejected. Otherwise, a warning message is output. The user then has the option to override the warning or to enter a new fd.

bb) batch mode

if the specified fd was used in the last GET command, the PRINT command is rejected and program execution is aborted. Otherwise, the file specified in fd is deleted and a "file-deleted" information message is output. The specified file is then allocated as described in (a) above.

When fd is omitted, the output is directed to the current list device.

EXAMPLES:

Example 1:

Output to the line printer the formatted document contained in FILE.A on the default volume.

```
Enter command:  >GET FILE.A

Enter command:  >PRINT PR:
```

Example 2:

Output a second copy of FILE.A to the line printer.

```
Enter command   >PRINT PR:
```

Example 3:

Make a copy of FILE.B on volume M300 to file FILE.PRT on the default volume. FILE.PRT does not exist.

```
Enter Command:  >GET M300:FILE.B

Enter Command:  >PR  FILE.PRT
```

Example 4:

Output a second copy of FILE.B to the current list device.

```
Enter command:  >PR
```

5.3.14 REPLACE Command

Mnemonic: REPLACE

Operands: R R=Range
none

Description: This command deletes existing text lines and inserts new text entered from the command input device. R specifies the lines to be deleted. If R is not specified, the current line is deleted. The new text is inserted in place of the first line deleted by the command. As many lines as desired can be inserted. OS/32 TEXT outputs a line number followed by the ">" symbol as a prompt to the user for the new text lines. Text input is terminated by depressing RETURN directly after the prompt is given.

Current Line: Last line entered.

EXAMPLES:

Example 1:

Replace lines 2-4 in current text.

```
Existing text:      1     * THIS TEXT IS PART OF
                    2     * A NEW DOCUMENT
                    3     *
                    4     * FIRST THING IS TO
                    5     * REVISE                    Current
                                                      Line

Enter command:  >REP 2-4

TEXT prompts:       2     >* AN OLD DOCUMENT          User enters:
                                                      text
                    2.01  >* PUB NO. 777              User enters:
                                                      text
                    2.02  >CR                         User enters:
                                                      Carriage return

Resulting Text:     1     * THIS TEXT IS PART OF
                    2     * AN OLD DOCUMENT
                    2.01  * PUB NO. 777               Current Line
                    5     * REVISE
```

Example 2:

Replace text in Current Line.

```
Enter command:  >REP                                  Range
                                                      is null

TEXT prompts:       2.01>* DOC NO. 40-372             User enters:
                                                      text
                    2.02>* FROM PROJ.Q FILE           User enters:
                                                      text
                    2.03>CR                           User enters:
                                                      Carriage Return

Resulting text:     1     * THIS TEXT IS PART OF
                    2     * AN OLD DOCUMENT
                    2.01  * DOC NO. 40-372
                    2.02  * FROM PROJ.Q FILE          Current Line
                    5     * REVISE
```

5.3.15 SAVE Command

Mnemonic: SAVE

Operands:		
	fd	fd=file descriptor
	*	*=current input file
	fd,R	R=Range
	*,R	

Description: Outputs text being edited to a file or device. If fd specifies a file which does not currently exist, an indexed file is allocated with the name fd. Upon completion of the SAVE command, the text output to fd is still available for further editing.

If fd is the name of an existing file, OS/32 TEXT does not save the text in that file in order to prevent accidental destruction of a master file. Instead, a warning message is issued and OS/32 TEXT awaits a user response. The user has the option to use the specified file or to specify a new fd.

The operand "*" may be used to specify the file that is currently being used as the input file. In this case, the file is overwritten with the text currently being edited.

The operand "R" specifies the text lines to be output. If "R" is omitted, all text lines are output.

Current Line: Last line output.

EXAMPLES:

Example 1:

Output entire text to "FILE.A" on default volume.

Enter command: >S FILE.A

Resulting text:	Entire text still available.	Current Line is last line in text.

Example 2:

Output lines 1-10 to FILE.B on volume M300.

Enter command: >S M300:FILE.B,1-10

Resulting text:	Entire text still available.	Current Line is line 10

Example 3:

Output entire text to the file which was used in the last "GET" command.

Enter command: >S *

Resulting text:	Entire text still available.	Current Line is last line in text.

5.3.16 SEND STOP Command

Mnemonic: SEND STOP

Operands: None

Description: This command allows the user to terminate the execution of lengthy commands and regain control of command input mode in OS/32 TEXT.

Commands affected by SEND STOP are: CHANGE, COLUMN, FIND, FORMAT, PRINT, and TYPE.

The command is only valid during interactive use of OS/32 TEXT and when the program is executing under either of the following conditions:

- as a foreground task, where OS/32 TEXT is the current task
- as a terminal task, under an MTM system generated to allow tasks to receive messages via the OS/32 SEND command

SEND STOP is entered in response to an OS or MTM prompt. A prompt is obtained by depressing the BREAK key, possibly several times.

Entering SEND STOP causes OS/32 TEXT to terminate processing of the command issued just prior to SEND STOP. After processing the SEND STOP command, OS/32 TEXT is ready to accept the next command.

5.3.17 TYPE Command

Mnemonic: TYPE

Operands: R R=Range
none

Description: Outputs one or multiple lines to the list device as specified by R. When R is omitted, only the current line is output.

Current Line: Last line output.

EXAMPLES:

Example 1:

Display all existing text lines.

```
Existing text:      1     AAAA
                    2     BBBB
                    2.01  CCCC                    Current Line
                    3     DDDD

Enter command:  >T1-

TEXT outputs:       1     AAAA
TEXT outputs:       1     BBBB
TEXT outputs:       2.01  CCCC
TEXT outputs:       3     DDDD                    Current Line
```

Example 2:

Display current line.

```
Enter command:  >T

TEXT outputs:       3     DDDD                    Current Line
```

Example 3:

Display line 2.01.

```
Enter command:  >T 2.01

TEXT outputs:       2.01  CCCC                    Current Line
```

5.3.18 Display Next Line(s)

Mnemonic: + (plus sign)
RETURN

Operands:		
	n	n is an integer
	none	

Description: Outputs the nth line following the current line to the list device. If the operand is omitted, the next line is output, as this is equivalent to the n=1. If the operand n is greater than the number of following lines in the text, then the last line of text is output with the information message "LAST LINE".

This command can be used to position the current line. It is especially useful for positioning to lines with duplicate line numbers which have resulted from the INSERT, REPLACE, or INCLUDE command.

Depressing RETURN directly following the prompt is equivalent to +1, and causes the next line to be displayed.

Current Line: The displayed line.

5.3.19 Display Previous Line(s)

Mnemonic: -(minus sign)

Operands:		
	n	n is an integer
	none	

Description: Outputs the nth preceding line in the file to the list device. If the operand is omitted, the preceding line is output. This is the equivalent of n=1. If the operand n is greater than the number of preceding lines in the text, then the first line is output, along with the information message "FIRST LINE".

This command can be used to position the current line. It is especially useful for positioning to lines with duplicate line numbers which have resulted from the INSERT, REPLACE, or INCLUDE command.

Current Line: The displayed line.

5.3.20 Line Number

Mnemonic: Line number

Operands: text
none

Description: Depending on its use, a "line number" provides the capabilities of the DELETE, INSERT, and REPLACE commands. The "line number" command actions are as follows:

- o Delete

  A line number without an operand deletes the specified line.

- o Replace

  A line number with a "text" operand replaces the specified line with the new text, if the line exists.

- o Insert

  A line number with a "text" operand inserts the specified line in the proper numeric position in the current text, if the specified line does not exist.

  The "line number" must be separated from the "text" by a space.

Current Line:

a) with "text" operand: specified line number

b) without operand: line following deleted line or the preceding line if the last line was deleted

CHAPTER 6
DESCRIPTION OF FORMATTING REQUESTS

6.1 INTRODUCTION

This chapter presents detailed information for each OS/32 TEXT formatting request. The material in this chapter is intended for reference use. The general usage of formatting requests is described in Chapter 4.

Section 6.1.1 provides a cross reference index by functions to the alphabetical listing of formatting requests in Section 6.3.

Section 6.2 explains the syntax of the formatting requests.

Section 6.3 contains descriptions of all formatting requests in alphabetical order.

Section 6.4 lists the initial values of formatting requests which have such a value.

6.1.1 Cross-Reference Index to Formatting Requests

The index lists formatting requests based on their functional relationship. The mnemonics for the formatting requests are indicated by capital letters.

- Define Page Size

  Line WIDTH
  Page LENGTH

- Margin Control

  MARGIN definition (absolute)
  Left MARGIN adjustment (relative)
  Right MARGIN Adjustment (relative)
  Top MARGIN
  Bottom MARGIN

- Text Line Filling

FILL output lines
do NOt FILL output lines
BREAK

- Text Line Positioning

  Left JUSTIFY
  Right JUSTIFY
  INDENTation
  new PARAGRAPH
  CENTERed text
  WIDOW PREVENTion

- Text Line Spacing

  line SPAC(E)ing
  SKIP Line
  page EJECT

- Page Titles

  page NUMBERing
  Top TITLE
  Bottom TITLE
  EVEN TOP Title
  EVEN BOTTOM Title
  ODD TOP title
  ODD BOTTOM title

- Footnotes and Tables

  FOOTNOTEs
  floating KEEP
  in context BLOCK

- Revision Identification

  revision BAR
  revision LEVEL

- Special Handling of Characters

  OVERPRINT
  special BLANK
  UNDERSCORE
  BOLD FACE
  HYPENAT(E)tion

- Date and Time Functions

  MONTH
  DAY
  YEAR
  time of DAY

- Text File Merging

  IMBED file

- User Interaction

  RETURN to user
  QUIT processing
  COMMENT
  change CONTROL character

6.2 Syntax

All formatting requests described in this chapter use the following format:

.request argument

where,

- the period "." is a control character generally used; if necessary the control character can be redefined by the .CONTROL formatting request;
- request is the name of the formatting request being described;
- the underlined part of the request and argument is the minimum acceptable abbreviation;
- the request is separated from the argument by at least one blank;
- argument may be a single character, a number, or a word.

Notations that describe arguments are:

- brackets [ ] indicate an optional argument
- braces { } indicate a choice of arguments

Formatting requests and their arguments may be entered in upper or lower case letters.

A formatting request must always be entered on a line by itself and must appear at the beginning of that line.

6.3 DESCRIPTION OF FORMATTING REQUESTS

This section presents the formatting requests in alphabetical order.

In the descriptions, the initial value is given for each formatting request if applicable.

The initial values are automatically reset each time the FORMAT command is entered to produce a formatted document. Section 6.4 contains a summary of initial values. The examples presented in the formatting request descriptions assume these initial values unless they are explicitly changed in an example.

For formatting requests which have optional arguments, the values which are substituted when arguments are omitted are given.

The descriptions also indicate whether or not the particular formatting request causes a "break" in the formatted output. Please refer to the description of the .BREAK formatting request for the definition of break.

6.3.1 Revision Bar

```
.BAR { BEGIN, number }
     { END          }
```

Initial Value: none

Break: yes

This formatting request is used to identify modifications to a document. The arguments BEGIN and END delimit the modified text. In the formatted output, the modified text is marked by revision bars. The argument "number" assigns a revision level number to the modified text. This revision level number is used in conjunction with the .RLEVEL formatting request to enable/disable the output of revision bars during formatting.

The placement of revision bars in the formatted output is subject to the following considerations:

- The revision bars appear separated by two (2) columns from the text area in either the left or right margin (see below). The margins must therefore be defined sufficiently wide enough to contain the revision bar.
- If no even/odd page titles are defined, the revision bar appears always in the right margin.
- If an even or odd page title is defined, the bar appears in the right margin on odd numbered pages and in the left margin on even numbered pages.

EXAMPLE:

Assume you have a document that was modified once and is at revision level 1 - and you want to add two modifications under revision level two (2).

```
Existing text
        .
        .
        .
Enter: .BAR BEGIN,2
       Modified Text Section 1
       .BAR END
        .
        .
        .
```

```
Enter: .BAR BEGIN,2
       Modified Text Section 2
       .BAR END
            .
            .
            .
```

Please see example in the .RLEVEL description for the procedure to control display of revision bars in the formatted output.

6.3.2 Special Blank

.BLANK [character]

Initial Value: $

Default: $ or character specified in last .BLANK request.

Break: no

The argument "character" specifies a single character. If the argument is omitted, the default value is assumed.

The next text line is scanned for the presence of "character". Each occurrence of that character is replaced by a blank (space). During formatting, the special blank character is considered a non-blank character. This means that text on either side of the special blank character is not spread out or split across two lines.

EXAMPLE:

Normally, two blanks are placed after each period in the formatted output text. This may not always be desirable in names and titles, for instance. The following example shows the use of a special blank in a name.

```
Enter: .BLANK
       My name is Dr.$A.$Phillips. I am a surgeon.

Result: My name is Dr. A. Phillips.  I am a surgeon.
```

6.3.3 In Context Block

.BLOCK { BEGIN / END }

Initial Value: none

Break: yes

This formatting request is used to define a block of text such as a table. The arguments BEGIN and END delimit the text within a block. "In context" means that the text in the block appears within the formatted output text at the same place where it was entered.

During the formatting process, the block is treated in its entirety to prevent splitting of the text between pages. The placement of the block is subject to the following considerations:

(a) If the current page contains text, the block is placed on this page only if the block fits entirely on the current page.

(b) If the current page contains text and the block does not fit entirely on this page, the balance of this page remains empty and the block is output starting at the top of the next page as described in (c) below.

(c) If the current page contains no text, output of the block is started and then continued on subsequent pages, if necessary, until all text in the block is output.

The placement criteria described in (b) above is the only difference in the processing of a block of text defined as an in context .BLOCK and a floating .KEEP.

The following restrictions apply to in context blocks:

- blocks cannot be nested
- blocks cannot contain floating keeps
- blocks cannot contain footnotes
- eject requests within the block are ignored

6.3.4 Bottom Margin

.BMARGIN lines

Initial Value: 5

Break: yes

This formatting request specifies in the argument the number of blank "lines" to occur at the bottom of each page. If a bottom title is specified, the bottom page title is centered vertically within this space.

When the bottom margin is odd, an equal number of blank lines appear above and below the title. When the bottom margin is even, one more blank line appears above the title than below the title.

6.3.5 Bold Face

.BOLDFACE

Initial Value: none

Break: no

This formatting request causes overprinting of all characters in the next text line to give them a darker appearance.

6.3.6 BREAK

.BREAK

Initial Value: none

Break: yes

The text lines immediately preceding and following this formatting request are not concatenated to fill output lines.

The description of each formatting request in this chapter indicates whether or not the request causes a break.

6.3.7 Bottom Title

.BIITLE /S1/S2/S3/

Initial Value: none

Break: no

This formatting request is used to specify a page title which appears at the bottom of each page in the space defined as the bottom margin.

The bottom title may consist of up to three independent sections as defined by the argument sections /S1/S2/S3/. All three sections appear on the same line in the bottom margin and are positioned with respect to the margin settings currently in effect as follows:

- o S1 is left justified
- o S2 is centered between the margins
- o S3 is right justified

Any of the three sections may be empty. Each section must be separated by a delimiting symbol. In the above description, a slash "/" is used. Any character, except a blank, may be used as the delimiter. The first character in the argument is assumed to be the delimiter.

The page number symbol may appear either in S1, S2, or S3. In the formatted output, the page number symbol is replaced with the computed page number for the current page.

EXAMPLES:

1. Specify a three-section bottom title with page number symbol in section number 3.

Enter: .BTITLE /TEXT/MANUAL/PAGE %/

Result:Assume page count equals 1.

TEXT MANUAL PAGE 1

2. Specify a two-section bottom title. Section two is empty. The delimiter is the asterisk.

```
Enter: .BT  *Volume I**No. 1-%*

Result:Assume page count equals 10.

       Volume I           No. 1-10
```

6.3.8 Centered Text

```
.CENTER { [BEGIN] }
        { [END]   }
```

Initial Value: none

Default: Only next text line is centered.

Break: yes

This formatting request causes text lines to be centered between the left and right margins currently in effect.

When the request is specified without an argument, only the next text line is centered.

To center a group of successive text lines, they must be delimited by the BEGIN and END arguments. When text is being centered, left and right justification and filling of output lines are suspended. This means that text lines appear in the formatted output exactly as they were entered - except they are centered.

6.3.9 Comment

.COMMENT note

Initial Value: none

Break: no

This formatting request allows the user to insert comments throughout the raw text to keep notes for personal use concerning the preparation, maintenance, and formatting requirements of the document. The argument "note" is any information that the user wants to insert.

The .COMMENT request is not processed during the execution of the FORMAT command and the "note" does not appear in the formatted text of the document.

6.3.10 Change Control Character

.CONTROL character

Initial Value: . (period)

Break: no

This formatting request allows you to change the control character. The control character is the first character of each formatting request.

Remember that each formatting request has to be entered on a line by itself and that it must appear at the beginning of that line.

If document text must be entered in such a way that a period is required at the beginning of a line, the formatting request control character can be changed. Once changed, the new character is in effect for all subsequently entered requests.

EXAMPLE:

You want to enter columns of fractions starting at the beginning of each line. You change the control character temporarily to a semicolon ";" and reset again to a period ".".

```
Enter: .BLOCK BEGIN
       .CENTER
       TEMPERATURE COEFFICIENTS
       .CONTROL ;
       ;NOFILL
       ;SKIP
       .0000   .0017   .0035   .0052   etc.
       .0175   etc.
       .0349   etc.
           .
           .
           .
       .2419   .2436    etc.
       ;BLOCK END
       ;CONTROL .
       .FILL
```

6.3.11 Day (from system)

.DAY [character]

Initial Value: none

Default: no character appended

Break: no

This formatting request reads the system clock and inserts the day of the month into the formatted output text. The day is represented as a one or two digit number.

The argument "character" specifies a single character which is appended to the day in the formatted output. This character is used primarily to add punctuation to the date.

Example:

This example shows how you can request the insertion of day, month, and year into the formatted document.

```
Enter: .LM +40
       123 Apricot Street
       Linton, New Jersey
       .MONTH
       .DAY ,
       .YEAR
       .LM RESET
       .SK 1
       Dear Sirs:
       .SK 1
       In this month of
       .MONTH
       the company
            ...
            ...
```

Result:

```
                                    123 Apricot Street
                                    Linton, New Jersey
                                    April 4, 1978

Dear Sirs:

In this month of April, the company ...
```

6.3.12 Page Eject

.EJECT

Initial Value: none

Break: yes

This formatting request causes termination of text output to the current page. The next text line appears on a new page. However, bottom margins, bottom titles and footnotes, if any of these are specified, are output to the current page.

If the current page is full when the .EJECT request is encountered, the request is ignored since OS/32 TEXT is about to output a new page.

6.3.13 Even Bottom Title

.EVENBOTTOM /S1/S2/S3/

Initial Value: none

Break: no

This formatting request is used to specify a page title which appears at the bottom of each even numbered page in the space defined as the bottom margin.

The bottom title may consist of up to three independent sections as defined by the argument sections /S1/S2/S3/. All three sections appear on the same line in the bottom margin and are positioned with respect to the current margin settings as follows:

- S1 is left justified
- S2 is centered between the margins
- S3 is right justified

Any of the three sections may be empty. Each section must be separated by a delimiting symbol. In the above description, a slash "/" is used. Any character, except a blank, may be used as the delimiter. The first character in the argument is assumed to be the delimiter. The page number symbol may appear in either S1, S2, or S3. In the formatted output, the page number symbol is replaced with the computed page number for the current page. For examples see .BTITLE.

The use of this request causes revision bars, as specified by the .BAR request, to appear in the right margin on odd numbered pages and in the left margin on even numbered pages.

6.3.14 Even Top Title

.EVENTOP /S1/S2/S3/

Initial Value: none

Break: no

This formatting request is used to specify a page title which appears at the top of each even numbered page in the space defined as the top margin (see .TMARGIN). The top title may consist of up to three independent sections as defined by the argument sections /S1/S2/S3/. All three sections appear on the same line in the top margin and are positioned with respect to the current margin settings as follows:

- S1 is left justified
- S2 is centered between the margins
- S3 is right justified

Any of the three sections may be empty. Each section must be separated by a delimiting symbol. In the above description, a slash "/" is used. Any character, except a blank, may be used as the delimiter. The first character in the argument is assumed to be the delimiter. The page number symbol may appear in either S1, S2, or S3. In the formatted output, the page number symbol is replaced with the computed page number for the current page. For examples, see .BTITLE.

The use of this request causes revisions bars, as specified by the .BAR request, to appear in the right margin on odd numbered pages and in the left margin on even numbered pages.

6.3.15 Fill Output Lines

.FILL

Initial Value: .FILL

Break: yes

This formatting request causes raw text lines to be concatenated to fill output lines during the formatting process.

To fill output lines words are collected from the raw text lines and assembled into output text lines according to the margin, indent, and justify specifications. Leading blanks are stripped from all words in the text line, and the occurrence of more than one blank between words is ignored. (See .INDENT and .NOFILL for exceptions to this rule.)

Words are placed in the output line until a word does not fit. This word becomes the first word of the next output line. When the first word which does not fit in the output line appears in the line following a .HYPHENATE request, an attempt is made to hyphenate that word.

A temporary suspension of the filling of output lines may be caused by a text line that contains only blanks or by a formatting request which causes a "break." The formatting request descriptions in this chapter indicate which requests cause a break. Note that a document text line consisting entirely of blanks will appear in the formatted output text, unless it occurs in a footnote.

After the output line has been filled in this manner, the line is adjusted as follows:

- When both left and right justify are in effect, multiple spaces are added between words so that the output line touches both margins. However, the last line in a paragraph is not spread.
- When only left or right justify is in effect, the output line is adjusted to touch the left or right margin only.
- When neither left nor right justify is in effect, the output line is left justified only.

Input lines correspond exactly to output lines only when .NOFILL is specified. Please see Chapter 4 for more information on output line filling.

6.3.16 Footnotes

.FOOTNOTE { BEGIN / END }

Initial Value: none

Break: no

This formatting request is used to define the text of a footnote. The arguments BEGIN and END delimit the text belonging to each footnote.

The footnote is output at the bottom of the current page. If not enough space is available on that page, output of the footnote is continued on subsequent pages until completed. The space allocated for footnotes on a page does not exceed one fourth of the text area available between the top and bottom margins.

The text of a footnote is separated from the document text by a dashed line.

Footnotes are processed subject to the formatting requests currently in effect. The following considerations apply:

- Footnotes are processed sequentially. That means, when multiple footnotes are encountered in short succession, the first footnote encountered is output completely before the second one, etc. Output of up to 10 footnotes may be pending at a time.
- Footnotes may not be nested.
- Footnotes may not contain .KEEP or .BLOCK.
- Footnotes are always single spaced.
- The .EJECT and .SPACE formatting requests are ignored if they are encountered in the text of a footnote.
- Blank lines in the text of a footnote do not appear in the formatted output.

6.3.17 Hyphenation

.HYPHENATE [character]

Initial Value: - (dash)

Default: - (dash) or character specified in last .HYPHENATE request.

Break: no

This formatting request affects only the next line of document text. The argument "character" is a hyphenation indicator which allows you to specify words to be conditionally hyphenated.

If the words containing the hyphenation indicators fit on the the current output line, they are output without hyphens. Otherwise, OS/32 TEXT attempts to use one of the indicated hyphens to hyphenate the last word on the line; all other hyphens in the word are not output.

To indicate conditional hyphenation in a word that normally contains a hyphen, the normal hyphen (dash) and the hyphenation indicator must be entered; e.g., tailor--made. If the indicator had been changed to an "@", you would enter tailor-@made. This assures that the mandatory hyphen in the word will always appear in the formatted output.

Please note that the above procedures apply only to words in the text line that is preceded by an .HYPHENATE request. That means that in any other text line, words that normally contain hyphens are entered without the hyphenation indicator; e.g., happy-go-lucky.

Remember, when entering text lines you do not hyphenate words at the end of a line as you would on a typewriter because the last character on one line and the first character on the next line are treated as separate words. See Chapter 4 (Entering Text).

EXAMPLES:

1. This example illustrates conditional hyphenation of words that are not normally written with a hyphen. The hyphenation character is changed.

```
Enter: .MARGIN 1,20
       .HYPHENATE @
```

```
        This is an excel@lent example of hyphena@tion
        tion of a word.

Result:This is an excellent
       example of  hyphena-
       tion of a word.
```

2. This example illustrates conditional hyphenation of words that are normally hyphenated. The hyphenation indicator is still the initial value: - (dash).

```
Enter: .MARGIN 1,21
       .HY
       My friend is a happy--go--lucky fel-low.

Result:My friend is a happy-
       go-lucky fellow.
```

3. This example illustrates the use of a hyphen without the use of the .HYPHENATE request.

```
Enter: .MARGIN 1,27
       Abraham Lincoln, 1809-1865, was the 16th
       president of the United States - from 1861-1865.

Result:Abraham Lincoln, 1809-1865,
       was the 16th  president  of
       the  United  States  - from
       1861-1865.
```

6.3.18 Imbed File

.IMBED fd

Initial Value: none

Break: no

The .IMBED formatting request allows the user to insert the contents of another file into the document currently being formatted. The argument "fd" specifies the name of the file (file descriptor) which contains the text to be inserted.

The file being imbedded may itself contain an .IMBED request for yet another file and so on up to a level of ten files. The file may contain either text or formatting requests or both.

Formatting of an imbedded file is terminated when end-of-file or a .QUIT formatting request is encountered. Formatting then continues with the text that contained the .IMBED request.

Please see Chapter 4 for application guidelines of the .IMBED request.

EXAMPLES:

1. Imbed the contents of the file named FILE.A.

```
Enter: This is IMBED-Example No. 1.
       .IMBED FILE.A
       End of Example No. 1.

Result:This is IMBED-Example No. 1.
       Text from FILE.A
       End of Examole No. 1.
```

2. Imbed the contents of files named FILE.B and FILE.C. FILE.B contains text, followed by an .IMBED request for FILE.C, and some more text.

```
Enter: This is IMBED-Example No. 2.
       .IMBED FILE.B
       End of Example No. 2.

Result:This is IMBED-Example No. 2.
       Text from FILE.B - part 1
```

```
Text from FILE.C
Text from FILE.B - part 2
End of Example No. 2
```

6.3.19 Indentation

.INDENT { [+] columns / - columns }

Initial Value: none

Break: yes

This formatting request defines a change in the left margin for the next text line only. The argument "columns" specifies how many columns (character positions) the left margin is to be adjusted from its current setting.

If the argument specifies a positive value, the margin is moved to the right; if negative, it is moved to the left.

A special case, outdenting, occurs when the margin is adjusted to the left. In this case, text appears to the left of the current left margin setting. Any text appearing in this area is not adjusted; meaning that leading spaces and multiple spaces between words are never deleted. This feature permits labelling of paragraphs. Any text appearing to the right of the left margin is subject to adjustment if .FILL is in effect.

EXAMPLES:

1. This is an example of indenting.

```
Enter: .MARGIN 10,35
       .INDENT 5
       The .INDENT request starts a new
       paragraph and adjusts the left margin
       of the first line in that paragraph.

Result:     The .INDENT request
       starts a new paragraph
       and adjusts the left
       margin of the first line
       in that paragraph.
```

2. This is an example of outdenting.

```
Enter: .MARGIN 10,50
       .INDENT -8
       6.3.19          INDENTATION
       The paragraph label 6.3.19 plus one space
       appear to the left of margin 10.  The
       leading spaces in front of INDENTATION
```

are removed because .FILL is in effect.

Result:6.3.19 INDENTATION
The Paragraph label 6.3.19 plus one space appear to the left of margin 10. The leading spaces in front of INDENTATION are removed because .FILL is in effect.

6.3.20 Floating Keep

.KEEP { BEGIN | END }

Initial Value: none

Break: no

This formatting request is used to define a block of text such as a table. The arguments BEGIN and END delimit the text within a block. "Floating keep" means that the placement of the block within the formatted output text may not always be in the same place where the block was entered.

The placement of the block is subject to the following considerations:

a) If the current page contains no text, output of the block is started and continued on subsequent pages, if necessary, until all text in the block is output.

b) If the current page contains text, the block is placed on this page only if it fits on it entirely.

c) If the current page contains text and the block does not fit entirely on this page, the balance of this page is filled with text which was entered after the block. The block is then output starting at the top of the next page as described in (a) above.

The placement criteria described in (c) above is the only difference from the processing of a block of text defined by .BLOCK.

Restrictions: Floating Keeps

- cannot be nested
- cannot contain .BLOCK
- cannot contain footnotes
- .EJECT requests within the defined block are ignored.

6.3.21 Left Justify

.LJUSTIFY [ON]
[OFF]

Initial Value: ON

Default: ON

Break: yes

When the ON argument is selected, this formatting request causes output lines to be justified against the left margin.

If the argument is omitted from the request, the ON condition is assumed. Left justification of output lines is subject to the following considerations:

- When both .LJUSTIFY and .RJUSTIFY are turned OFF, the output lines are left justified.
- Left justification is temporarily suspended by the following formatting requests:

  .CENTER

  .INDENT

  .PARAGRAPH

6.3.22 Line Width

.LWIDTH columns

Initial Value: 85

Break: yes

This formatting request defines the physical width of the page that will contain the formatted document. The argument "columns" expresses the width in character positions that fit on a line.

The left and right margins are offsets within this limit.

The initial value corresponds to an 8 1/2" paper width.

6.3.23 Left Margin Adjustment (Relative)

```
.LMARGIN { [+] columns }
         {  -  columns }
         { RESET       }
```

Initial Value: none

Break: yes

This formatting request allows you to make margin adjustments relative to the left margin currently in effect.

The argument "columns" specifies the number of character positions the margin is to be adjusted. The margin can be adjusted to the left (-) or to the right (+). The new margin remains in effect until changed again.

Each time a margin adjustment is made, OS/32 TEXT records the left margin setting that was in effect prior to the adjustment. A total of ten (10) margin adjustments can be recorded by OS/32 TEXT.

The RESET argument is used to return the left margin setting to its previous setting.

Multiple adjustments or resets may be performed in succession. Each request corresponds, in reverse order, to its equivalent adjustment request. If more requests are entered than adjustments were made, the margins remain at the current setting.

The recording table for the margin adjustment is cleared each time the FORMAT command is issued or the .MARGIN request is processed.

The following example illustrates the use of .LMARGIN.

EXAMPLE:

Left and right margins are defined followed by multiple adjustments and reset requests.

Enter		Result
.MARGIN	16,66	Margins are 16,66 (Text area occupies 17-65)
.LMARGIN	+3	Margins are now: 19,66 (Text area occupies 20-65)
.RM	-3	Margins are now: 19,63 (Text area occupies 20-62)
.LM	-10	Margins are now: 9,63 (Text area occupies 10-62)
.RM	RESET	Margins are now: 9,66 (Text area occupies 10-65)
.LM	RESET	Margins are now: 19,66 (Text area occupies 20-65)

6.3.24 Margin Definition (Absolute)

.MARGIN left margin,right margin

Initial Value: 12,74

Break: yes

This formatting request sets the absolute left and right margins of a page.

The arguments represent character positions or columns on the physical page, counted from left to right. Their value must be within range of the line width defined by the .LWIDTH formatting request.

The arguments "left margin" and "right margin" define the number of blank columns which appear to the left and to the right of the text area respectively.

The margins provided as the initial values are the commonly used settings for page size 8 1/2" x 11".

Generally, the margins set by this request remain in effect throughout the formatting of a document, providing a reference base for temporary changes in the margin settings.

The formatting requests .LMARGIN and .RMARGIN allow you to make margin adjustments relative to the settings defined by .MARGIN. The formatting requests .INDENT and .PARAGRAPH allow you to temporarily override the left margin setting to facilitate indenting and outdenting of a text line.

EXAMPLE:

```
.LWIDTH  80
.MARGIN   15,66
```

The resulting eighty character text line has fifteen blanks, followed by fifty characters of text, followed by fifteen blanks.

6.3.25 Month (from system)

.MONTH [character]

Initial Value: none

Default: no character appended

Break: no

This formatting request reads the system clock and inserts the current month into the formatted output text.

The months are represented by the words: January, February, etc.

The argument "character" specifies a single character which is appended to the month in the formatted output. This character is used to add punctuation to the date. Please see .DAY for an example.

6.3.26 Do Not Fill Output Lines

.NOFILL

Initial Value: .FILL

Break: yes

This formatting request inhibits the effect of .FILL.

This means that:

- formatted output text lines are not filled by concatenating new text lines;
- leading spaces are not removed from words;
- multiple spaces between words are not removed.
- the output lines are adjusted to touch the current left margin if
  - left justify is in effect
  - both left and right justify are in effect, or
  - both left and right justify are OFF
- the output lines are adjusted to touch the current right margin if right justify is in effect.

The result is that formatted output lines correspond exactly to raw text lines except for the effects of margin settings, centering, and left/right justification requests as listed above.

A typical application of this request is to prepare text in table form where rows and columns of information must appear exactly as they were entered by the user.

It is important to remember that when NOFILL is in effect, an input line cannot contain more characters than space is available in the output line. The available space in the output line is determined by the line width specification minus the current margin settings.

EXAMPLE:

```
.LWIDTH 80
.MARGIN 10,70
.NOFILL
Available Space:  59 spaces
```

6.3.27 Odd Bottom Title

.ODDBOTTOM /S1/S2/S3/

Initial Value: none

Break: no

This formatting request is used to specify a page title which appears at the bottom of each odd numbered page in the space defined as the bottom margin (see .BMARGIN). For details please refer to the description of .EVENBOTTOM.

6.3.28 Odd Top Title

.ODDTOP /S1/S2/S3/

Initial Value: none

Break: no

This formatting request is used to specify a page title which appears at the top of each odd numbered page in the space defined as the top margin (see .TMARGIN). For details please refer to the description of .EVENTOP.

6.3.29 Overprint

.OVERPRINT [character]

Initial Value: /

Default: / or character specified in last .OVERPRINT request.

Break: no

This formatting request causes overprinting of a specified character in the text line directly following the request.

The argument "character" defines the overprint indicator. If the argument is omitted, the overprint character is the default value.

The next line is scanned for the presence of the indicator. Each occurrence of the indicator causes the character directly preceding it to be overwritten by the character following the indicator.

EXAMPLES:

1. The default character is the overprint indicator.

```
Enter: .OVERPRINT
       Sen/~orita

Result:Señorita
```

2. Change the overprint indicator to a ":".

```
Enter: .OV :
       GO:/TO:/

Result:GØTØ
```

3. Partially underline a word.

```
Enter: .OVERPRINT /
       F/_O/_RMAT

Result:FORMAT
```

6.3.30 New Paragraph

.PARAGRAPH [[+] columns]
[- columns]

Initial Value: 0

Default: 0 or column count specified in the previous .PARAGRAPH request.

Break: yes

This formatting request is used to indicate the start of a new paragraph. The request defines a change in the left margin for the next text line. It also causes the new paragraph to be separated from preceding text by a number of blank lines equal to the multiple spacing count (see .SPACE) plus one (1) line.

The argument "columns" specifies how many columns or character positions the left margin is to be adjusted from its current setting.

If the argument specifies a positive value, the margin is moved to the right; if negative, it is moved to the left.

If the argument is omitted, the margin is adjusted according to the default value. The default value is either the initial value of "0" or the value specified in the previous .PARAGRAPH request.

A special case, outdenting, occurs when the margin is adjusted to the left. In this case, text appears to the left of the current left margin setting. Any text appearing in this area is not adjusted, meaning that leading blanks and multiple blanks between words are not deleted. This feature permits labelling of paragraphs. Any text appearing to the right of the left margin is subject to adjustment if .FILL is in effect.

The examples given for the .INDENT request, to illustrate indenting and outdenting, are equally valid for .PARAGRAPH.

6.3.31 Page Length

.PLENGTH lines

Initial Value: 66

Break: yes

This formatting request defines the physical length of the page that will contain the formatted document. The argument "lines" expresses the length in number of lines that fit on the page.

The top and bottom page margins are offsets within this limit. The initial value corresponds to a page length of 11 inches.

6.3.32 Page Numbering

.PNUMBER [initial] [,[increment][,character]]

Initial Value: initial = 1
increment = 1
character = %

Default: initial = 1 at the start of formatting or the current calculated page count during formatting

increment = 1 or the value specified in last .PNUMBER request

character = % or the character specified in last .PNUMBER request

Break: no

This formatting request initializes automatic page numbering.

The argument "initial" specifies the page number that appears on the first formatted output page following the request. If the argument is omitted, the page is numbered according to the current default value.

The argument "increment" specifies the increment by which the page numbering counter is advanced after each page output. If the argument is omitted, the increment is the default value.

The argument "character" specifies a page number symbol. This symbol may be specified in the various page title requests available in OS/32 TEXT (see .BTITLE for an example). In the formatted output, this symbol is replaced with the current page number.

If the argument is omitted, the current default character is the page number symbol.

EXAMPLES:

Request	Explanation
.PNUMBER 1,2,@	Pages are numbered starting with number 1, in odd numbers: 1,3,5,7 etc.

To print the page number, the "#" must appear in place of the page number in a page title request.

.PNUMBER 10,,!

Assume the current default for "increment" = 1.
Pages are numbered, starting with number 10, in increments of one: 10,11,12,13,etc.
To print the page number, the "!" must appear in place of the page number in a page title request.

.PNUMBER ,5

Assume twelve numbered pages have been output when this request is encountered and that the current page count = 13.

Pages are numbered starting with number 13, in increments of 5: 13,18,23,etc.
The page number symbol remains unchanged.

6.3.33 Quit Processing

.QUIT [ALL]

Initial Value: none

Default: see text below

Break: no

This formatting request causes processing of the current text input to be terminated. If the argument ALL is specified, all processing is terminated and control is returned to the OS/32 TEXT command mode.

If the argument is omitted, input termination and subsequent action depend on where the .QUIT request is encountered, as follows:

- When .QUIT is encountered in an "imbed" file (see .IMBED), input from that file is terminated and control returned to the source of the .IMBED request.
- When .QUIT is encountered during .RETURN processing, control is returned to the source of the .RETURN request.
- When .QUIT appears in the text processed by the FORMAT command, all processing is terminated and control returned to the OS/32 TEXT command mode. The text processed by the FORMAT command is the text that was entered using APPEND, etc. or retrieved from a disc file using the GET command.

6.3.34 Return to User

.RETURN [comment]

Initial Value: none

Default: no comment displayed

Break: no

During the formatting process (i.e., when the FORMAT command is being executed) control can be switched temporarily to the command input device by using the .RETURN formatting request. At that time, the user can enter document text and formatting requests at the command input device. This input is processed immediately and becomes part of the formatted output. Input is continued until terminated by entering the .QUIT formatting request or depressing RETURN as the only input on a line.

Text entered under the control of the .RETURN request is not inserted into the raw text which contains the .RETURN request.

The argument "comment" may contain alphanumeric user information. This information is displayed on an interactive command input device only. It is intended to provide the user prompt regarding the type of data to be entered.

.RETURN cannot be used if the same device is used for command input and formatted output.

6.3.35 Right Justify

.RJUSTIFY [ON]
[OFF]

Initial Value: ON

Default: ON

Break: yes

When the ON argument is selected, this formatting request causes output lines to be justified against the right margin.

If the argument is omitted, the ON condition is assumed. Right justification of output lines is subject to the following considerations:

- When both .LJUSTIFY and .RJUSTIFY are ON, right justification is not done if .NOFILL is in effect.
- The last line of a paragraph, or the line preceding a formatting request which causes a break, is not right justified when both .LJUSTIFY and .RJUSTIFY are ON and .FILL is in effect.
- Right justification is temporarily suspended by the .CENTER formatting request.

6.3.36 Revision Level

.RLEVEL number [,[left column][right column]]

Initial Value: number=0
left column=10
right column=76

Default: left column - 2 columns to the left of current left margin
right column - 2 columns to the right of the current right margin

Break: no

This formatting request controls the output and placement of revision bars for text areas entered with the .BAR request.

The argument "number" refers to revision level numbers entered with .BAR. If "number" is greater than any revision level in the text, no revision bars are output. If "number" is not greater, all revisions equal to or greater than "number" are identified with revision bars.

The arguments "left column" and "right column" specify the column number in the left and right margins respectively in which the revision bars appear in the formatted document.

If the arguments are omitted, the revision bar columns are computed to be 2 columns to the left for "left column" and 2 columns to the right for "right column" respective of the left and right margin settings which are in effect when the .RLEVEL request is encountered.

EXAMPLES:

1. Assume three revision levels (1,2,3) exist in your text. No revisions are to be identified.

Enter: .RLEVEL 4

Result: No revision bars output.

2. Assume same text as in (1) above. Revision level 3 is to be identified.

Enter: .RLEVEL 3

Result: All text entered under .BAR 3 is identified with revision bars.

3. Assume same text as in (1) above. All .BAR revisions made to the text are to be identified.

Enter: .RLEVEL 1

Result: All text entered under .BAR 1 through 3 is identified with revision bars.

6.3.37 Right Margin Adjustment (Relative)

```
.RMARGIN { [+] columns }
         {  -  columns }
         { RESET       }
```

Initial Value: none

Break: yes

This formatting request allows you to make margin adjustments relative to the right margin currently in effect.

The argument "columns" specifies the number of character positions by which the margin is to be adjusted. The margin can be adjusted to the left (-) or to the right (+). The new margin remains in effect until changed again.

Each time a margin adjustment is made, OS/32 TEXT records the right margin setting that was in effect prior to the adjustment. A total of ten (10) margin adjustments can be recorded by OS/32 TEXT.

The RESET argument is used to return the right margin setting from an adjustment to its previous setting. Multiple adjustments or resets may be performed in succession. Each reset request corresponds, in reverse order, to its equivalent adjustment request.

If more RESET requests are entered than adjustments were made, the margins remain at the current setting.

The recording table for the margin adjustments is cleared each time the FORMAT command is issued or the .MARGIN request is processed. Please refer to .LMARGIN for examples.

6.3.38 Skip Line

.SKIP [lines]

Initial Value: none

Default: 1

Break: yes

This request causes blank lines to appear in the formatted output. The argument "lines" specifies how many blank lines are to be output. If the argument is omitted, one blank line is output. The .SKIP request is subject to the following considerations.

- If the end-of-page is encountered before all blank lines specified in a request have been output, output of the remaining lines is inhibited.
- If the output of blank lines would occur at the top of a page, output of these lines is inhibited.
- If .SKIP appears within a footnote it is ignored.

6.3.39 Multiple Spacing

.SPACE lines

Initial Value: 1

Break: yes

This request sets the spacing of text lines. The argument "lines" specifies the line spacing:

1 = single spacing

2 = double spacing

3 = triple spacing

The value of "lines" must be within the range of 1 through 6.

The .SPACE request affects only the text area in the formatted output.

Text in footnotes is always single spaced.

6.3.40 Time of Day (from system)

.TIME [character]

Initial Value: none

Default: no character appended

Break: no

This formatting request reads the system clock and inserts the time of day into the formatted output text.

The time is represented as an eight character string specifying hours, minutes, and seconds as follows: HH:MM:SS.

The argument "character" specifies a single character which is appended to the time in the formatted output. This character is used to add punctuation to the time.

EXAMPLE:

Insert the time in the output and specify a sentence terminator.

```
Enter: This example was done at
       .TIME .

Result:  This example was done at 11:23:56.
```

6.3.41 Top Margin

.IMARGIN lines

Initial Value: 5

Break: yes

This formatting request specifies the number of blank lines to occur at the top of each page.

When specified, the top page title is centered vertically within this space. When the top margin is an odd number of lines, an equal number of blank lines appear above and below the title. When the top margin is an even number of lines, one more blank line appears above the title than below the title.

6.3.42 Top Title

.IITLE /S1/S2/S3/

Initial Value: none

Break: no

This formatting request is used to specify a page title which appears at the top of each page in the space defined as the top margin.

The top title may consist of up to three independent sections as defined by the argument /S1/S2/S3/. All three sections appear on the same line in the top margin and are positioned as follows:

- o S1 is left justified
- o S2 is centered between the margins
- o S3 is right justified

Any of the three sections may be empty. Each section must be separated by a delimiting symbol. In the above description, a slash "/" is used. Any character, except a blank, may be used as the delimiter. The first character in the argument is assumed to be the delimiter.

The page number symbol (see .PNUMBER) may appear in S1, S2, or S3. In the formatted output, the page number symbol is replaced with the computed page number for the current page.

Please see .BTITLE for examples.

6.3.43 Underscore

.UNDERSCORE

Initial Value: none

Break: no

This formatting request causes alphanumeric characters in the next text line to be underscored.

Non-alphanumeric characters (punctuation characters, parentheses, brackets, etc.) or parts of a word can be underscored using the .OVERPRINT request. To underscore blanks, simply enter the underscore character in place of the blank.

EXAMPLE:

```
Enter: This is an example of
       .UNDERSCORE
       underscoring
       in OS/32 TEXT.
```

Result: This is an example of underscoring in OS/32 TEXT.

6.3.44 Widow Prevention

.WIDOWPREVENT [ON]
[OFF]

Initial Value: ON

Default: ON

Break: yes

This formatting request affects the placement of the first and last line of a paragraph in the formatted output.

The request is subject to the following considerations:

- When .FILL and .WIDOWPREVENT ON are selected, the first line of a paragraph does not appear as the last line on a page, nor does the last line in a paragraph appear as the first line on a page.
- When .WIDOWPREVENT OFF is selected, all paragraph lines are treated as regular text lines.
- When .WIDOWPREVENT ON and .NOFILL are selected, all paragraph lines are treated as if .WIDOWPREVENT OFF had been selected.

6.3.45 Year (from system)

.YEAR [character]

Initial Value: none

Default: no character appended

Break: no

This formatting request reads the system clock and inserts the year into the formatted output. The year is represented as a four digit number.

The argument "character" specifies a single character which is appended to the year in the formatted output. This character is used to add punctuation. Please see .DAY for an example.

6.4 TABLE OF INITIAL VALUES

Many formatting requests have initial values which are set up automatically each time the FORMAT command is entered. These values are intended to provide a standard format so that you can obtain formatted output with minimum initial setup. If these initial values are not suitable, they can, of course, be changed.

The following table lists initial values of formatting requests which have such values.

Formatting Request	Initial Value	Comments
.BLANK	$	Special blank indicator
.BMARGIN	5	Bottom page margin = 5 lines
.CONTROL	.	Formatting request control character
.FILL	.FILL	Filling of output lines enabled.
.HYPHENATE	-	Hyphenation indicator
.LJUSTIFY	ON	Left justification of output lines enabled.
.LWIDTH	85	Physical page width = 85 columns (or character positions). 85 corresponds to an 8 1/2" paper width.
.MARGIN	12,74	Left margin = 12, right margin = 74 columns (or character positions) counted from left to right on the physical page (see .LWIDTH above).
.NOFILL	.FILL	Filling of output lines enabled.
.OVERPRINT	/	Overprint indicator
.PARAGRAPH	0	No left margin adjustment of first line in paragraph.
.PLENGTH	66	Physical page length = 66 lines. This corresponds to an 11" page length.

Formatting Request	Initial Value	Comments
.PNUMBER	1	FirstPage number
	1	Page number increment
	%	Page number symbol
.RJUSTIFY	ON	Rightjustification of output lines enabled.
.RLEVEL	0	Output revision bars for all document revisions made with the .BAR request of revision level 0 and greater.
	10	Column number to place revision bars in left margin
	76	Column number to place revision bars in right margin
.SPACE	1	Single line spacing in effect
.TMARGIN	5	Top page margin = 5 lines
.WIDOWPREVENT	ON	When FILL is in effect, first and last lines of paragraphs do not appear on a different page, separated from body of paragraph.

CHAPTER 7
USE OF TEXT-II

7.1 GENERAL

This chapter explains the use of OS/32 TEXT-II, the version that contains only formatting capability, as explained in Chapter 1.

The chapter assumes:

- That you understand the use of TEXT-I as described in this manual.

- That you know how to use OS/32 EDIT as described in the OS/32 EDIT User's Manual, Publication No. 29-516. Please note that the difference in usage between OS/32 EDIT and OS/32 TEXT-I is the special TEXT-I commands, FORMAT and PRINT, which are not available in OS/32 EDIT. These commands are recognized and processed by TEXT-II.

7.2 CAPABILITIES

The capabilities of TEXT-II are as follows:

- Generates formatted output from document text and formatting requests contained in a file which was previously created with OS/32 EDIT.

- Prints a copy of a formatted document which is contained in a file and was previously formatted with TEXT-II.

- TEXT-II recognizes only the following commands:

  END
  FORMAT
  GET
  PAUSE
  PRINT
  SEND STOP

The use of these commands is the same as described for TEXT-I. Please note that before the FORMAT and PRINT commands can be used, the text file to be processed must be made available to TEXT-II via the GET command.

Loading, starting, and task establishment are identical to the procedures described in Chapter 2 for TEXT-I. To apply the descriptions in Chapter 2 to TEXT-II, simply substitute the appropriate file name where reference is made to TEXT32.

Following are two examples illustrating the combined use of OS/32 EDIT and TEXT-II. The examples use the simplest forms of loading and starting these programs and apply to an OS/32 MTM environment. The "*" and ">" symbols shown are MTM system and TEXT-II prompts respectively, indicating that they are ready to accept a user response.

7.2.1 Example 1

Enter document text and formatting requests into OS/32 EDIT and save the text and formatting requests in a file named EXAMPLE.1. Use TEXT-II to obtain formatted output of the document text saved in file EXAMPLE.1.

User/Computer Dialogue	Description
*	System ready to accept a command
*LOAD EDIT32.TSK	User loads EDIT32
*START	User starts EDIT32
OS/32 EDIT Rn	EDIT32 displays identification
>	EDIT32 ready to accept a command
>APPEND	User requests text input mode
1 > EXAMPLE NO. 1 2 > .SKIP . > . . > . . > RETURN	User enters document text and formatting requests User terminates text input mode

User/Computer Dialogue	Description
>	EDIT32 ready to accept a command
>SAVE EXAMPLE.1	User saves text in file
>END	User terminates EDIT32
USER-ID -END OF TASK CODE=0	EDIT32 terminating message
*	System ready to accept a command
*LOAD TEXT32FM.TSK	User loads TEXT-II
*START	User starts TEXT-II
OS/32 TEXT-II Rn	TEXT-II displays identification
>	TEXT-II ready to accept a command
>GET EXAMPLE.1	User makes document text to be processed available to TEXT-II
>FORMAT	User requests formatted output of document to default list device
>FORMAT EXA1.FMT	User requests a second formatted output of the document to a file named EXA1.FMT.
>END	User terminates TEXT-II
USER-ID -END OF TASK CODE=0	TEXT-II terminating message
*	System ready to accept a command

7.2.2 Example 2

Use TEXT-II to obtain a copy of the document text which was formatted and output to file EXA1.FMT in Example 1.

User/Computer Dialogue	Description
*	System ready to accept a command
*LOAD TEXT32FM.TSK	User loads TEXT-II
*START	User starts TEXT-II
OS/32 TEXT-II Rn	TEXT-II displays identification
>GET EXA1.FMT	User makes formatted document text to be processed available to TEXT-II.
>PRINT	User requests a copy of the formatted document to be output to the default list device.
>PRINT PR:	User requests a second copy of the formatted document to be output to the printer.
>END	User terminates TEXT-II.
USER-ID -END OF TASK CODE=0	TEXT-II terminating message
*	System ready to accept a command

CHAPTER 8
ERROR HANDLING

8.1 INTRODUCTION

This chapter lists and describes the messages output by TEXT.

Section 8.2 contains the messages related to the use of TEXT commands.

Section 8.3 contains the messages resulting from the use of TEXT formatting requests as they are processed by the FORMAT command.

8.2 COMMAND ERROR MESSAGES

This section lists messages which are output during processing of TEXT commands.

In interactive mode, message output is directed to the command device. Following the message output, TEXT is in command mode awaiting the next user command.

In batch mode messages are output to the list device. The first error message encountered causes the execution of TEXT to be terminated.

The messages are divided into the following two groups:

a) Syntax error messages, which are caused by an improper command format;

b) Execution error messages, which are caused when TEXT cannot complete the execution of a command as specified by the operands.

Within each group, the messages are listed in alphabetical order. The messages are based on the operation of the OS/32 EDIT program. The descriptions given here are a summary of the chapter on USER MESSAGES contained in the OS/32 EDIT User's Manual, Publication No. 29-576. You may refer to

this manual to supplement the information provided in this section.

8.2.1 Syntax Errors

All syntax errors messages are preceded by a question mark (?). A syntax error message is generated when TEXT cannot recognize the command, or the operand.

The current line is not affected by a syntax error.

MESSAGE	DESCRIPTION
?COLUMN # INCORRECT	Column number appended to string operand is invalid.
?COL WITHIN STRING OPERAND INVALID HERE	A column specification for the replacement string is not meaningful.
?COMMA MISSING	A comma is expected in the syntax but not found.
?COMMAND NOT RECOGNIZED	An invalid command mnemonic has been entered.
?FILE NAME INCORRECT	Format of required file name invalid. See Chapter 5 for proper file descriptor format.
?FILE NOT FOUND	Syntax of the file descriptor is correct, but specified file could not be found on the specified volume.
?LINE OPERAND INVALID	A line specification is not a line number or string operand.

?INVALID COLUMN SPECIFIER	The column number is an invalid number or exceeds the number of columns in the current text.
?NUMBER INVALID	An operand might be a line number or an integer, but syntax is incorrect.
?RANGE INVALID	Range operand is not recognized. Line specification may be wrong. Separator is not a minus sign.
?STRING INVALID	String operand incorrectly specified. Trailing delimiter might be missing, or delimiter might not be an acceptable character.

8.2.2 Execution Messages

All execution messages are preceded by an exclamation point (!). These messages are generated when TEXT cannot complete the execution of a command as specified by the operands. The command may have been partially executed on some of the specified text lines. The TYPE command without an operand can be used to identify the "current line" at which execution of the command may have stopped.

Execution messages which indicate a more serious problem are denoted by two exclamation points. In some cases, the only way to recover is to attempt to save the current text, (on a different file from the source for GET), and then to end the editing session and restart.

MESSAGE	DESCRIPTION
!DUPLICATE START OPTION FOUND	An option is entered more than once.
!!FILE ERR, LUn,r,fd	where: n=logical unit number r=reason fd=file descriptor These messages are output when OS/32 TEXT cannot resolve an

SVC7 error internally.

The possible reasons and corresponding SVC7 error status bytes are:

REASON	SVC 7 ERROR STATUS
ILLEGAL FUNCTION	01
LU ERROR	02
VOLUME ERROR	03
NAME ERROR	04
SIZE ERROR	05
PROTECT ERROR	06
PRIVILEGE ERROR	07
BUFFER ERROR	08
ASSIGNMENT ERROR	09
TYPE ERROR	0A
FILE DESCRIPTOR	0B
TGD ASSIGNMENT ERROR	0C
ACCOUNT ERROR	0D

Refer to the OS/32 Program Reference Manual, Publication Number 29-613, for SVC7 error status bytes.

MESSAGE	DESCRIPTION
!INVALID BLOCK SIZE	The value specified for the BLOCK operand exceeds 255, or is invalid.
!INVALID COMMAND DEVICE	The device (or file) specified in the COMMAND option of the START command is synactically incorrect.
!INVALID HEX CHAR	A specified string while the Hexadecimal option is in effect contains an illegal hexadecimal digit.

!INVALID LIST DEVICE	The device (or file) specified in the LIST option of the START command is syntactically invalid.
!INVALID START OPTION	TEXT does not recognize an option appearing in the START command.
!!I/O ERR,LUn,r,fd	where: n=logical unit number r=reason fd=file descriptor These messages are output when OS/32 TEXT cannot resolve an SVC1 error internally. The possible reasons and corresponding SVC1 error status bytes are:

REASON	SVC 1 ERROR STATUS
ILLEGAL OR UNASSIGNED LU	X'81'
PARITY OR RECOVERABLE ERROR	X'82'
UNRECOVERABLE ERROR	X'84'
END OF FILE	X'88'
END OF MEDIUM	X'90'
DEVICE UNAVAILABLE	X'A0'
ILLEGAL FUNCTION	X'C0'

Refer to the OS/32 Program Reference Manual, Publication Number 29-613, for a detailed explanation of the SVC1 error status bytes.

MESSAGE	DESCRIPTION
!LENGTH OPTION NOT ACCEPTED	The LENGTH operand of the OPTION command is only acceptable when no text is available.
!LINE NOT FOUND	Specified line not present in text.
!LINE TOO LONG	The line of text just entered while Tab Conntrol is in effect or a line being changed is too long; command processing

	stops and the line which generates the message is not accepted.
!LIST OPTION OMITTED-BATCH MODE	If the COMMAND option specifies a non-interactive device, TEXT assumes execution in batch mode. For batch mode execution, the LIST option is required in the START command.
!NO TEXT	No text is available for editing.
!OPTION ERROR FOLLOWS XXXX	Option operand is not recognized; XXXX are the last four non-blank characters of the command that were recognizable. Command processing stops at the last valid operand.
!OPTION HEX NOT ALLOWED HERE	1. Hexadecimal mode is recognized only for commands that alter or display existing text. 2. Option HEX cannot be selected if there is no current text.
!RANGE 1 NOT FOUND	The first line specification of the Range operand cannot be located in existing text.
!RANGE 2 NOT FOUND	This error message results when the second line specification of the Range operand cannot be located in existing text.
!SYNTAX ERROR IN START OPTINS	The options in the START command are incorrectly entered.
!TABS TABLE FULL	No more tab points can be accepted. The maximum number of tab points TEXT can handle is 20.
!TAB ILLEGAL	Tab option incorrectly specified. Character given may be illegal.
!UNABLE TO ASSIGN LIST DEVICE	The device (or file) specified in the LIST option of the START command cannot be assigned.

8.3 FORMATTING ERROR MESSAGES

This section lists messages which are output as a result of problems encountered during the formatting of document text as requested by the FORMAT and PRINT commands.

All messages are sent to the list device in both interactive and batch modes.

Formatting error messages are divided into the following two groups:

a) Error Messages, which are caused by formatting request syntax errors and improper formatting request usage, prevent the continuation of the formatting process. Following output of an error message in interactive mode, TEXT returns to the command mode and awaits the next user command. In batch mode, execution of TEXT is terminated.

b) Warning messages, which result from improper use of formatting requests but whose effect is less severe, permit the formatting process to continue. In both interactive and batch modes, formatting continues after output of a warning message.

Error and warning messages have the following format:

```
TEXT DIGNOSTIC AT LINE nnn FILE fd
ERROR XYY     message
```

or

```
TEXT DIAGNOSTIC AT LINE nnn FILE fd
WARNING XYY     message
```

where:

nnn is the number of the input text line which caused the message output. This number corresponds to the line number assigned to this text line by TEXT during the editing process. It can therefore be used to locate the line in the raw document text for error correction, etc.

Note, however, that when the text being formatted contains fractional line numbers (e.g., 1.01,5.07, etc.) it should first be SAVEd in a file and then retrieved with the GET command. This procedure will renumber the text lines and synchronize them with the line numbers in the messages.

fd is a file name or a device mnemonic. If the text being formatted is contained in a file, fd is the name of the file which contains the line that caused the error.

If the text being formatted was entered directly from the command input device (keyboard, card reader), fd represents the device mnemonic of that device, e.g., CON:, CR:, etc.

X refers to one of the following message categories:

0	syntax error
1	general diagnostic
2	heading or footing diagnostic
3	page dimension diagnostic
4	page or line filling diagnostic
5	block diagnostic
6	word diagnostic
7	function diagnostic

YY refers to the message number within a category.

message = is a brief description of the error condition.

8.3.1 Error Message

In the following sections, error and warning messages are listed in numerical order by categories and error numbers.

8.3.1.1 Category 0: Syntax Errors

ERROR001 INVALID FORMATTING REQUEST

This error may be caused by the following conditions relating to the specification of a formatting request:

- the request is misspelled or not all characters for the minimum abbreviation have been entered;
- the request is not separated from its first argument by at least one blank;

- a document text line begins in column one with the control character reserved for formatting requests. The CONTROL formatting request allows re-definition of the control character. See Chapter 5.

ERROR002 INVALID ARGUMENT

A formatting request contains an invalid argument. Verify the proper argument specification for the affected formatting request.

ERROR003 MISSING ARGUMENT

A mandatory argument has been omitted from a formatting request. Verify the proper argument specification for the affected formatting request.

8.3.1.2 Category 1: General Errors

ERROR101 INVALID NESTING OF INPUT FILES

This error is caused by too many levels of input nesting with the IMBED formatting request. Ten (10) files maximum may be simultaneously open for imbedding. To remedy this condition, combine text from several files into one file to reduce the level of input nesting.

ERROR102 NONEXISTENT PAGE NUMBER

A page number specified in the FORMAT command does not exist in the document being formatted.

ERROR103 NO INPUT TEXT

This error is caused when the FORMAT and PRINT commands are entered and no text is available for processing. To use the PRINT command, text must first be made available to TEXT via the GET command. To use the FORMAT command, text must first be entered into TEXT from the command device or retrieved from a file via the GET command.

ERROR104 DATA INTEGRITY CHECK

This error can occur when using the PRINT command to print a previously formatted file. It is possible that the user tried to manipulate the data within the file and hence the file cannot be processed by the PRINT command.

ERROR105 IMBED FILE ERROR fd

An attempt was made to assign the file fd as an Imbedded File. The assignment failed. The file name fd could be mistyped, or assigned exclusively to another run.

8.3.1.3 Category 2: Errors in Page Headings and Footings

ERROR201 TITLE SPECIFICATION IN ERROR

An error occurred in the title specification of a TITLE or BTITLE formatting request. It is usually caused by inconsistent use of or missing title delimiters. Refer to Chapter 6 for proper format to specify top and bottom page titles.

8.3.1.4 Category 3: Page Dimension Errors

ERROR301 BOTTOM MARGIN ERROR

The bottom margin specified in the BMARGIN formatting request is a negative number or greater than the difference between the defined page length (PLENGTH) and the top margin (TMARGIN). The bottom margin must be in the following range:

$$0 \geq \text{BMARGIN} \geq (\text{PLENGTH}-\text{TMARGIN})$$

ERROR302 TOP MARGIN ERROR

The top margin specified in the TMARGIN formatting request is a negative number or greater than the

difference between the defined page length (PLENGTH) and the bottom margin (BMARGIN). The top margin must be in the following range:

$$0 \leq TMARGIN \leq (PLENGTH-BMARGIN)$$

ERROR303 MARGIN SPECIFICATION ERROR

The margin settings specified in the MARGIN formatting request are in error for one of the following reasons:

- o both margins specify the same setting;
- o the left margin setting is greater than the right margin setting;
- o either one or both of the margin settings specifies a setting that falls outside the physical page width as defined by the line width (LWIDTH) formatting request.

The margin settings must be in the following range:

$$0 \leq \text{Left Margin} < \text{Right Margin} \leq LWIDTH$$

ERROR304 LINE WIDTH ERROR

The number specified in the LWIDTH formatting request is outside the legal range. The line width must be in the range:

$$1 \leq LWIDTH \leq 132$$

ERROR305 PAGE LENGTH ERROR

The number specified in the PLENGTH formatting request is outside the legal range. The page length must be in the range:

$$1 \leq PLENGTH \leq 65536$$

8.3.1.5 Category 4: Errors Caused by Filling Pages and Lines

ERROR401 WIDOW PREVENTION ERROR

The argument specified in the .WIDOWPREVENT formatting request is not one of the two legal keywords - ON, OFF - for this request.

ERROR402 MULTIPLE SPACING ERROR

The number specified in the SPACE formatting request is outside the legal range. The multiple space count must be in the range:

$$1 \leq \text{SPACE} \leq 6$$

ERROR403 INDENT SPECIFICATION ERROR

The number specified in the INDENT request causes an adjustment to the current left margin which places the start of the next line outside the boundaries of the physical output page.

8.3.1.6 Category 5: Errors Caused by Blocks and Footnotes

ERROR501 ILLEGAL NESTING OF BLOCKS, KEEPS, OR FOOTNOTES

This error is caused by attempting to nest the formatting requests which define Blocks, Floating Keeps and Footnotes. Use of these requests within themselves or within each other is illegal.

The error also occurs when the END delimiter of a previously started Block, Keep or Footnote request was omitted and a second BEGIN delimiter for any one of these requests is encountered.

ERROR502 TOO MANY FOOTNOTES, BLOCKS, OR KEEPS

Many footnote, block, or keep formatting requests have been entered in close succession. The document should be changed to make these structures larger or space these formatting requests farther apart.

ERROR503 INSUFFICIENT MARGIN FOR REVISION BAR

The RLEVEL formatting request specifies use of the default column positions for the revision bars on the output page but there is insufficient room on the margin to place the oars.

The default positions for the revision bars are computed to be two (2) columns to the left and right respective of the left and right margin settings which are in effect when the RLEVEL request is encountered. If one of the resulting revision bar positions falls outside the physical output page, this error message is generated.

ERROR504 MAXIMUM NUMBER OF FOOTNOTES EXCEEDED

This error is caused when output of ten (10) footnotes is pending and another footnote request is encountered.

To alleviate this problem, try to spread out the definition of footnotes to provide sufficient output pages so that TEXT can complete the output of each footnote as it is encountered.

8.3.1.7 Category 6: Word Diagnostics

ERROR601 ILLEGAL OVERPRINTING

The overprint request is being used to overprint one character more than three times. Change the input text to avoid this.

8.3.2 Warning Messages

8.3.2.0 Category 0: Syntax Warnings

WARNING001 INVALID ARGUMENT

Extraneous characters occur on the request line. The request was accepted and the valid arguments were processed. The line should be checked for errors.

8.3.2.1 Category 1: General Warnings

WARNING101 THE .RETURN REQUEST WAS ISSUED ILLEGALLY

Document text and formatting requests are currently being entered under the control of the RETURN request. Another RETURN request cannot be entered at this time.

8.3.2.2 Category 2: Warnings in Page Headings and Footings

WARNING201 INSUFFICIENT SPACE FOR PAGE NUMBER

The text for the title sections of a Top or Bottom page title does not leave sufficient room, after the sections are positioned on the output page, to accommodate all digits for the page number.

8.3.2.3 Category 3: Page Dimension Warnings

WARNING301 INVALID MARGIN SET OR RESET

More than ten margin changes were specified without doing a reset. A margin reset request was encountered without a preceding margin adjustment request. The current margin settings remain unchanged.

8.3.2.4 Category 4: Warnings Caused By Filling Pages and Lines

WARNING401 CENTERING ALREADY IN EFFECT

This warning message is output when two successive CENTER BEGIN formatting requests are encountered without an intervening CENTER END request.

WARNING402 INPUT TOO LARGE FOR OUTPUT LINE

This error occurs when filling of output text lines is inhibited - NOFILL is in effect - and the current input line is longer than the available space in the output line. The current margin settings determine the available space in the output line. The input line will be truncated in the formatted output.

8.3.2.5 Category 5: Warnings Caused by Blocks and Footnotes

WARNING501 REVISION BAR BLOCKING IS ERRONEOUS

Text to be identified with revision bars must be defined by BAR BEGIN and BAR END formatting requests. This warning is issued when:

- o two (2) successive BAR BEGIN requests are encountered without an intervening BAR END request, or
- o a BAR END request is encountered without a preceding BAR BEGIN request.

WARNING502 BADLY STRUCTURED BLOCK, KEEP, FOOTNOTE, OR CENTER

This message is issued when one of these formatting requests contains a BEGIN argument and is not terminated by a request containing an END argument, or when an END argument is used when no previous BEGIN argument is used.

8.3.2.6 Category 6: Warnings Caused by Words and Characters

WARNING601 NO OVERPRINT CHARACTER APPEARS IN THIS LINE

An OVERPRINT request was encountered but the following text line does not contain the currently defined overprint symbol.

WARNING602 NO SPECIAL-BLANK CHARACTER APPEARS IN THIS LINE

A BLANK formatting request was encountered but the following text line does not contain the currently defined special blank symbol.

WARNING603 NO HYPHENATION CHARACTER APPEARS IN THIS LINE

A HYPHENATE formatting request was encountered but the following text line does not contain the currently defined hyphenation symbol.

APPENDIX A
COMMAND SUMMARY

The underlined letters in each command are the minimum abbreviation required. Commands may be entered in upper or lower case.

MNEMONIC	OPERANDS*	CURRENT LINE	COMMENTS
APPEND	none	last line entered	
CHANGE	S1,S2,R	last line of range	
COLUMN	C,S,R	last line of range	
DELETE	R	line following last line deleted	
END			
FIND	S,R	last line of range	
FORMAT	fd,page, HALT,XREF		All operands can be null
GET	fd	first line of fd	
INCLUDE	L,fd,R	last line inserted	L,fd can be null
INSERT	L	last line inserted	L can be null
OPTION	BLOCK=n LENGTH=n LIST=fd LOG=fd MODE=ASCII or HEX NOLOG NOTAB NOVERIFY TAB=char,stop,stop.. TERMINATOR=char VERIFY	Unchanged for all operands	default values: BLOCK=5 LENGTH=80 LIST=determined by START command MODE=ASCII NOLOG NOTAB TERMINATOR= unspecified VERIFY

PAUSE		unchanged	
PRINT	fd	not applicable	fd can be null
REPLACE	R	last line inserted	
SAVE	fd,R	last line output	R=null=entire file
SEND STOP		last line processed	Operating System command
TYPE	R	last line of range	
+	n	line output	n=null=1=carriage return: next line is output
-	n	line output	n=null=1: previous line is output
line no.	text none	line number referenced line following deleted line	

---

* S = string operand
R = range operand
C = column number

fd = file descriptor
L = L operand = S or line no.
n = integer

FORMATTING REQUEST SUMMARY

Underlined letters in each formatting request and argument represent the minimum abbreviation required. Arguments in brackets are optional; arguments in braces indicate alternate choice.

The Initial Values are automatically set up each time the FORMAT command is entered.

The Default Values are substituted when an optional argument is omitted.

A formatting request which causes a break interrupts filling output lines when .FILL is in effect.

Formatting requests and arguments may be entered in upper or lower case.

REQUEST & ARGUMENTS	INITIAL VALUE	DEFAULT VALUE	BREAK
.BAR BEGIN,number END	none	N/A	Yes
.BLANK [character]	$	$ or character specified in last .BLA	No
.BLOCK BEGIN END	none	N/A	Yes
.BMARGIN lines	5	N/A	Yes
.BOLDFACE	none	N/A	No
.BREAK	none	N/A	Yes
.BTITLE S1/S2/S3/	none	N/A	No
.CENTER BEGIN END	none	next text line only	Yes
.COMMENT note	none	N/A	No
.CONTROL character	.(period)	N/A	No

.DAY [character]	none	no character appended	No
.EJECT	none	N/A	Yes
.EVENBOTTOM /S1/S2/S3/	none	N/A	No
.EVENTOP /S1/S2/S3/	none	N/A	No
.FILL	.FILL	N/A	Yes
.FOOTNOTE BEGIN END	none	N/A	No
.HYPHENATE [character]	-(dash)	-(dash) or character specified in last .HY	No
.IMBED fd	none	N/A	No
.INDENT [+]columns - columns	none	N/A	Yes
.KEEP BEGIN END	none	N/A	Yes
.LJUSTIFY ON OFF	ON	ON	Yes

REQUEST & ARGUMENTS	INITIAL VALUE	DEFAULT VALUE	BREAK
.LLENGTH columns	80	N/A	Yes
.LMARGIN [+]columns - columns RESET	none	N/A	Yes
.MARGIN left,right	12,74	N/A	Yes
.MONTH [character]	none	no character appended	No
.NOFILL	.FILL	N/A	Yes
.ODDBOTTOM /S1/S2/S3/	none	N/A	No
.ODDTOP /S1/S2/S3/	none	N/A	No
.OVERPRINT [character]	/	/ or character specified in last .OV	No
.PARAGRAPH [+]columns - columns	0	0 or column count specified in last .PA	Yes
.PLENGTH lines	66	N/A	Yes
.PNUMBER [initial] [,[increment] [,[increment]]	initial=1 increment=1 character=%	initial=1 at start of formatting or the current calculated page count during formatting increment=1 or value character=% character specified in last .PN	No
.QUIT [ALL]	none	Return control to: 1. Source of IMBED request, or 2. Source of RETURN request, or 3. TEXT command input	No
.RETURN [comment]	none	no comment displayed	No
.RJUSTIFY ON OFF	ON	ON	Yes
.RLEVEL number [,[left col]	number=0 left col=10	N/A left col=2 columns to the	No

[,right col]]	rght col=76	left of current left margin right col=2 columns to the right of current right margin	
.RMARGIN [+]columns - columns RESET	none	N/A	No
.SKIP [lines]	none	1	Yes
.SPACE lines	1	N/A	Yes
.TIME [character]	none	no character appended	No
.TMARGIN lines	5	N/A	Yes
.TITITLE /S1/S2/S3/	none	N/A	No
.UNDERSCORE	none	N/A	No
.WIDOWPREVENT ON OFF	ON	ON	Yes
.YEAR [character]	none	no character appended	No

APPENDIX B
TASK ESTABLISHMENT

The following sequence can be used to establish OS/32 TEXT as a segment task from an object version named TEXT.OBJ:

```
*LOAD TET32
*ASSIGN 5,CON:
*ASSIGN 7,CON:
*START
>ESTABLISH TASK,PURE
>INCLUDE TEXT.OBJ
>EXPAND n
>BUILD TASK,TEXT32
>END
```

The use of the EXPAND command provides work space memory for TEXT which may improve response time when editing text consisting of many lines in very large files.

TThe argument "n" is expressed as a number of 256 byte blocks of memory that is added to the task's Impure area. It should have a minimum value of 1 to provide space for START parameters.

The size of the workspace created by the TET32 EXPAND command may be overridden at load time. The "seg-size increment" argument in the LOAD command overrides the amount of work soce memory allocateed for the task by TET. The seg-size may be specified in .25KB increments.