Volume 2
Working with Text and Spreadsheets

OSBORNE™

COMPUTER CORPORATION

VOLUME 2

Working With Text and Spreadsheets

OSBORNE
EXECUTIVE
GUIDES

Chapter 1, *WordStar,* was written by Michael McCarthy.
Heather Allen, Andrew Emery, Melanie Allen, Ellen Lockett, and Suzanne Kopple contributed to *SuperCalc.*

Joseph Caggiano was editorial advisor.

Part No. 3F00194-00

Osborne Computer Corporation reserves the right to make changes or improvements to the equipment, software, and documentation herein described at any time and without notice.

Notice: Please be advised that any tampering with the internal hardware of the Osborne Executive Computer may void the warranty.

FCC Notice

This equipment generates and uses radio frequency energy. If not installed and used properly, that is, strictly according to the manufacturer's instructions, the equipment may cause interference with radio and television reception.

This equipment has been type tested and found to comply with the limits for a **Class B** computing device as specified in FCC Rules, Part 15, Subpart J, which is designed to provide reasonable protection against such interference in a residential installation.

If the equipment does cause interference to radio or television reception, which can be determined by turning it ON and OFF, try to correct the interference by doing one or more of the following:

Reorient the receiving antenna.
the computer with respect to the receiver.
Move the computer away from the receiver.
Plug the computer into a different outlet so the computer and receiver are on different branch circuits.

If necessary, the user should consult the dealer or an experienced radio/television technician for additional suggestions. The user may find the following booklet prepared by the Federal Communications Commission helpful:

> "How to Identify and Resolve Radio-TV Interference Problems".

This booklet is available from the U.S. Government Printing Office, Washington, D.C. 20402, Stock No. 004-000-00345-4.

To maintain Class B compliance use properly shielded and grounded cables when connecting this equipment to peripheral devices.

Warning

This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of FCC Rules. Only peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class B limits may be attached to this computer. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.

Contents

Print Controls 97

File Handling 117

Printing Files 132

Displaying 263

Editing and Protecting 275

Working with Larger Blocks 281

Part 4

CBASIC

Part 5

Microsoft BASIC

Index

Introduction

This volume deals with WordStar, MailMerge, SuperCalc, CBASIC, and MicroSoft BASIC. Each part is independent of the others, so you may start anywhere in the book. We recommend, however, that you read through WordStar before using MailMerge. If this is your first exposure to word processing or electronic spreadsheets, Volume 0, the *Beginner's Guide,* offers a first lesson in each which requires no previous knowledge. Volume 1, *Mastering the Osborne Executive,* provides an introduction to the Osborne Executive Computer.

The information in each part of this book is divided in three levels: basic, intermediate, and advanced. The basic level deals with general information and the often used commands. It assumes no previous knowledge other than the introductory chapters in the *Beginner's Guide*. The intermediate level takes you through additional commands and options which will increase your knowledge of the programs. The advanced level assumes a familiarity with the programs. It contains advanced commands and more complicated uses of the program.

Do not work with your master disks. These are a valuable investment. If you have not already done so, make copies of these and store your originals in a safe place. Both *Mastering the Osborne Executive* and the *Beginner's Guide* have information on copying these disks and on formatting data disks for storing your work.

We have supplied an installation appendix for WordStar and an appendix containing information on hooking up the most common printers.

PART 1
WordStar

Introduction

WORDSTAR is the popular word processing program for microcomputers.

It's fast, easy to use, and displays helpful information directly on screen. And WordStar is very powerful, providing you with many options for customizing word processing to individual needs.

IF YOU'RE NEW to word processing, we recommend you start with the *Osborne Executive Beginner's Guide*. It will introduce you to word processing, and give you a valuable overview of WordStar's writing, editing and printing capabilities. The *Beginner's Guide* assumes no previous knowledge of computers or word processing programs. It's a good introduction to WordStar and to the **Osborne Executive** computer.

If you have some familiarity with word processing, this WordStar Guide will teach you WordStar as quickly as you like, to whatever depth you need, giving you new information as required.

The topics in this guide are arranged in order of use: commands for starting the program; then commands for creating documents; commands for editing and correcting; special page-layout commands; and finally, printing commands. In this way, WordStar's features can be learned and tried in a logical sequence.

This WordStar Guide doesn't ask you to learn everything about the program before using it.

WordStar has nearly 200 commands, but most users need only a few dozen to become WordStar experts! And these few commands are the ones most easily learned and remembered.

So we've organized WordStar into three learning levels:

Basic Level, defined **The Basic Level**, explains simply and thoroughly, all the WordStar commands needed to use WordStar on a day-to-day basis. The only assumption is that you've tried the *Beginner's Guide* if you're completely new to computers.

Intermediate Level, defined **The Intermediate Level**, takes you further into specific groups of WordStar commands. Intermediate Level commands add to WordStar's power and versatility. Use this level to add optional, highly convenient commands, shortcuts and "Quick Tips" to your WordStar repertory.

Advanced Level, defined **The Advanced Level**, assumes thorough familiarity with WordStar, and some knowledge of aspects of computing such as CP/M. Once you're comfortable with WordStar, you'll want to review Advanced Level commands for special variations that may apply to your own work.

How to Use This Guide

1. If you haven't used a computer before, start with the *Beginner's Guide*. It will introduce you to WordStar and show you how to prepare disks for use.

   If you are somewhat familiar with computers, review *Volume 1–Mastering the Osborne Executive*, which explains operation of the computer and the underlying CP/M features.

2. Prepare a working copy of your WordStar disk and format several data disks. The procedures are explained in both the *Beginner's Guide* and *Mastering the Osborne Executive.*
3. Turn ON the **Osborne Executive**, insert the WordStar copy in drive **A** and the data disk in drive **B**. Close both drive doors and press RETURN.
4. Select the group of commands in this WordStar Guide which interest you. Practice the commands as you read about them.

This guide is organized in sections by function, from startup through printing of a final document. Each command group progresses from Basic through Advanced level. You can select the group and level of commands according to your needs.

MAILMERGE

The **MailMerge** section follows WordStar. MailMerge is a related program for generating mailing lists and form letters with WordStar.

WORDSTAR REFERENCE GUIDE

Volume 5 contains the *WordStar Reference Guide.* This lists WordStar commands in alphabetical order by functional group (Block Commands, Quick Commands, etc.).The *WordStar Reference Guide* has formal definitions of WordStar commands. This is a convenient way to check exact details of commands.

Before Editing

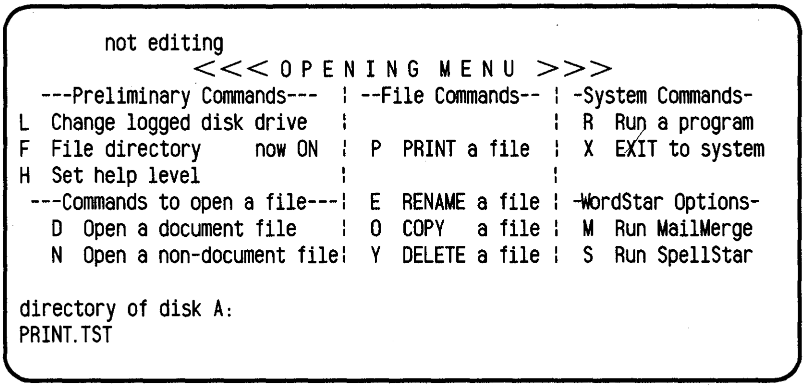
Basic Level

Logging the Disk
Naming Files
Open Document File

The Opening Menu

This section describes those options on the Opening Menu that allow you to start a file and begin editing.

The Opening Menu, Screen 1, is displayed on the screen when WordStar is started.



```
      not editing
             <<< O P E N I N G  M E N U >>>
  ---Preliminary Commands---  ¦ --File Commands-- ¦ -System Commands-
L  Change logged disk drive   ¦                   ¦  R  Run a program
F  File directory     now ON  ¦  P  PRINT a file  ¦  X  EXIT to system
H  Set help level             ¦                   ¦
 ---Commands to open a file---¦  E  RENAME a file ¦ -WordStar Options-
   D  Open a document file    ¦  O  COPY    a file ¦  M  Run MailMerge
   N  Open a non-document file¦  Y  DELETE a file ¦  S  Run SpellStar

directory of disk A:
PRINT.TST
```

SCREEN 1. Opening Menu

This menu lists those commands you can issue to WordStar at this time.

You will be using two of these commands now: Change Logged Disk Drive (L) and Open a Document File (D).

The other commands on the Opening Menu are explained in the File Handling section.

Logging the Disk

L *Change Logged Disk Drive*

The first thing you should do upon starting up WordStar is change the "logged disk drive" from drive **A** to drive **B**. We'll show you how.

You'll find the phrase "directory of disk A:" in the middle of the screen, just below the Opening Menu (see Screen 1). This Directory names the files available on the "logged disk drive," which is always drive **A** when WordStar is first started.

The "logged" drive is the drive WordStar reads files from and saves files to. You can specify otherwise each time you open a file, but for convenience you will usually prefer to change your "logged drive" over from drive **A** to drive **B**.

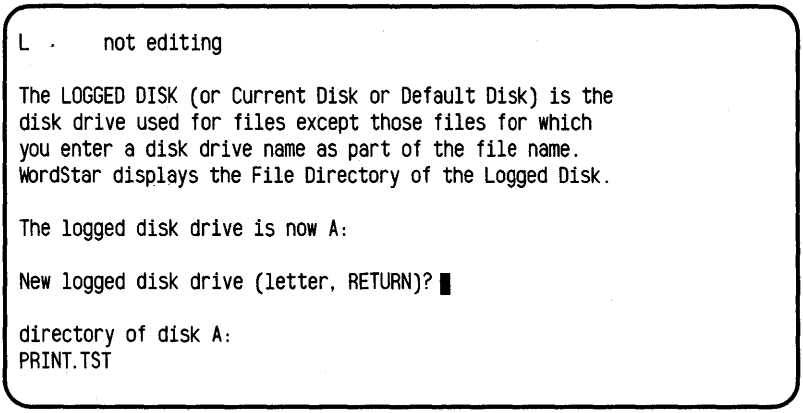
You "log in" drive **B** so you can keep your data files there, separate from your programs on drive **A**. Data files are any documents (letters, memos, reports, and other text) that you type into the computer. Program files, like WordStar and SuperCalc, contain computer instructions. They are usually stored in computer code, not in readable text. For ease of use and organizational convenience, program disks are usually put into the **A** drive and data disks in the **B** drive.

Here's how to change the logged drive.

The first of the "Preliminary Commands" on the Opening Menu on the screen is "L - Change logged disk drive." Issue this command as follows:

Press L.

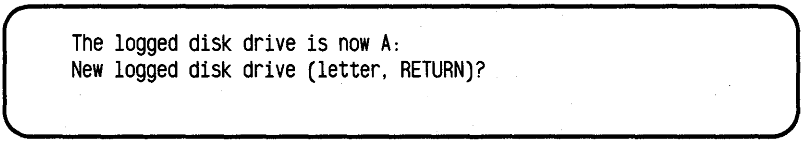
Notice how WordStar responds: (See Screen 2)



```
L  ·    not editing

The LOGGED DISK (or Current Disk or Default Disk) is the
disk drive used for files except those files for which
you enter a disk drive name as part of the file name.
WordStar displays the File Directory of the Logged Disk.

The logged disk drive is now A:

New logged disk drive (letter, RETURN)? █

directory of disk A:
PRINT.TST
```

SCREEN 2. Changing Logged Disk Drive

First, the screen is cleared of the Opening Menu. (The Directory remains on the screen.) A brief reminder of how the L command works is displayed; below that, this instruction appears:



```
    The logged disk drive is now A:
    New logged disk drive (letter, RETURN)?
```

At the end of this sentence the CURSOR waits for your response. (The cursor is the flashing block that shows your position on screen. Whatever you type appears on the screen at the cursor.)

Press [B]. The letter "B" appears on the screen at the cursor.

Press [RETURN].

The Opening Menu now re-appears as in Screen 1; but this time it displays the Directory for disk B.

If you are using a blank, newly formatted data disk, no file names will appear yet. As you add data files, they will be shown in this directory.

(There are certain kinds of program files called SYSTEM FILES which do not show up on the Directory listing. The CP/M manual explains system and directory files. There may be "hidden" program files on a disk even though the Directory shows nothing. Thus labeling your disks is important. The CP/M command "SHOW" will examine your disk files in more detail. We explain this command in "Seeing How Much Room You Have On Your Disks," later in this WordStar Guide.)

Use the "L" logging command each time you put a new disk in drive B so you can see what data files are on the new disk.

Starting or Editing a File

[D] *Open A Document File*

To start "processing words" you need to open a file for them. A file is the computer equivalent of a file folder. It saves a document or portion of a document on the disk so you can use it later or print it out.

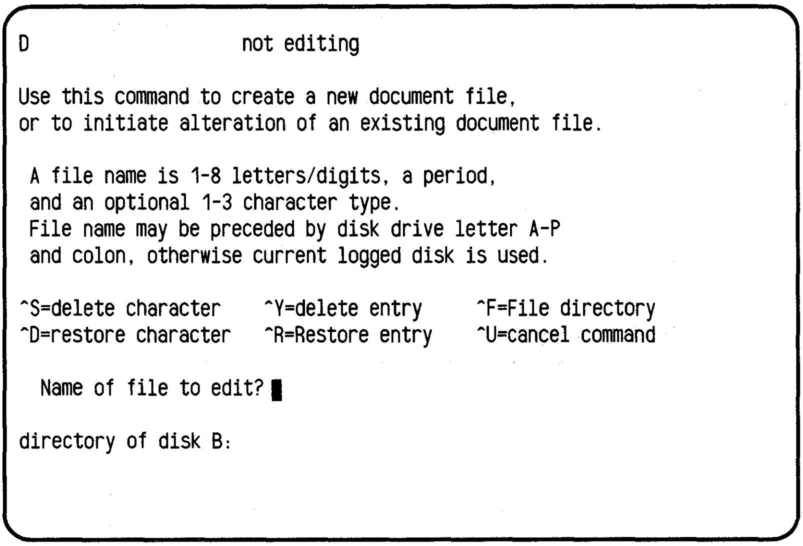
The "D" command in the Opening Menu tells WordStar you plan to start writing a document, or that you wish to edit a document you've already written. WordStar considers anything you write to be part of a document.

Here is how to begin a document file or to edit an existing document file.

Look at the Opening Menu on screen and notice the command "D Open a document file" under the group "Commands to open a file." To issue this command and begin a document:

Press D.

WordStar clears the Opening Menu from the screen.



```
D                    not editing

Use this command to create a new document file,
or to initiate alteration of an existing document file.

 A file name is 1-8 letters/digits, a period,
 and an optional 1-3 character type.
 File name may be preceded by disk drive letter A-P
 and colon, otherwise current logged disk is used.

^S=delete character    ^Y=delete entry     ^F=File directory
^D=restore character   ^R=Restore entry    ^U=cancel command

  Name of file to edit? █

directory of disk B:
```

SCREEN 3. D Opens a Document File

An explanation of files and file names appears on the screen to remind you of the file rules. There is also a list of ways to make corrections when entering the file name.

At the end of the explanations, WordStar asks:

```
        Name of file to edit?
```

- If you are starting a **new** document, type its name following the naming rules described below.
- If you are **editing** an existing file, type the name of the file to be edited.

When you've typed the name, which will appear on screen as you type it, check to see that you spelled the file name correctly. The easiest way to correct errors is to use the Left Arrow Key to back up to them. Then type the correction. Other commands for editing are listed on the screen for advanced users.

When the correct name is typed in, press RETURN.

If You Change Your Mind

If you CHANGE YOUR MIND about wanting to start a file, use the INTERRUPT COMMAND ^U (hold down the CTRL key and press U). This must be done BEFORE you press RETURN.

The computer will check the logged disk for the file name. If it is present, the file with that name is brought to the screen for editing.

If no file with that name is present, the phrase NEW FILE appears briefly on screen. WordStar opens a new file under the name you gave, and a blank screen is made available for your use.

Saving a file and quitting your WordStar session while writing a practice document are explained later in this guide. However, if you must stop your WordStar session before getting to the file saving section, here are two important commands you can use for now: SAVE and QUIT.

To SAVE the document you are working on, use the command ^KD. Hold down the CTRL key and press K; release both keys and press D. Your document will be saved onto the disk under the name you gave when you started.

To QUIT the document, use the command ^KQ. Hold down the CTRL key and press K; release both keys and press Q. If you have already written something on the file, answer Y to the question "Abandon Edited File?" You will be returned to the Opening Menu (Screen 1) and nothing you typed will be saved, not even the document's name. This is also a good way to abandon a document you've changed your mind about editing, because the original version will be left on the disk unchanged.

If you wanted to edit an existing file and instead got a new file, you may have misspelled the file name. Quit the document with the ^KQ command described above so you can start over.

Rules for Naming Files

Here are some simple rules for naming your data files while using WordStar.

A. The name can be from 1 to 8 letters or numbers long, upper or lower case. The name need not mean anything, but it should be useful to you.

B. There are only four small Don't's for file names:

- don't use a name that already exists on that disk;
- don't use a period (.) as the first character;
- don't use a question mark (?);
- don't use an asterisk (*).

If you use a period as the first letter, or a question mark or asterisk somewhere in the name, WordStar says "Invalid file name" and waits for you to try again.

Any other symbols on the keyboard can be used in making a file name.

C. AFTER the one-to-eight-character file name, you can put a period and then add an "extension" of up to three more characters/numbers. Don't use an asterisk or question mark or a second period in this extension. Examples of extensions are:
.LTR, .REP, .DOC, .TXT, .FIL, .OLD and .NEW.

Examples of Valid File Names

1423/55
MIKE.FIL
TASK-ONE.LS%
12.3
SAMPLE.TXT
DDENSITY.#32
LETTER
FEB28.LTR
GAS$-1&2

Don't use the extension .BAK because WordStar uses it. Other programs like BASIC and CP/M use extensions to identify special files they need. These are discussed in the Intermediate Level. You can always use the extensions .TXT, .DOC, and .LTR safely without problems.

Before Editing

Intermediate Level

Naming Files
BDOS Errors
Changing Disks

More on Naming Files

Some programs actually need specific kinds of file name extensions to work. *.COM* for example, is used to signify a program. If you type the first part of a .COM file name while in CP/M, the computer will attempt to load and run the program. *.BAS* is used for BASIC programs and *.ASM* to indicate assembly language programs.

WordStar creates two particular extensions that you may notice from time to time:

.$$$ indicates a temporary file that WordStar left behind, perhaps during a power failure or some other kind of program problem. It is just a filename in the directory and contains nothing and may be deleted.

.BAK is the backup copy WordStar makes automatically of any file you edit. This backup copy is available to help you reconstruct your file should you lose it for any reason.

When you edit an existing file, WordStar renames the old file by appending the extension .BAK, substituting it for the existing extension, if any.

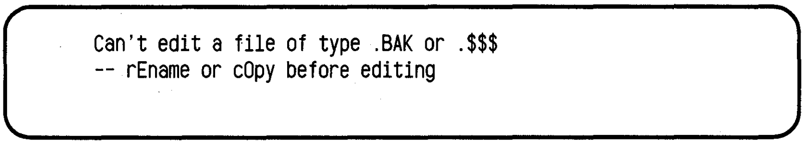
When you file your edited version, it has the correct name, and the previous version has the .BAK extension. Any earlier .BAK versions are deleted.

Later in this manual we'll describe how to save your files often so the backup copy at any moment is a recent version of the current file. This simplifies the process of recovering from an operational problem.

Two files which differ only in the extension, such as MAR.LTR and MAR.DOC, will have only one backup file, MAR.BAK. This will be the backup of whichever file was most recently edited.

How To Use .BAK Files

If you try to access, print or copy a .BAK file, WordStar will tell you:



```
Can't edit a file of type .BAK or .$$$
-- rEname or cOpy before editing
```

.BAK files can't be edited or printed. They can be copied to make a second file under a different name, or read into another file using the ^KR command, because this still leaves the original backup copy intact.

Backup copies are intended for backup. To use one, you must first RENAME it to remove the .BAK extension, or COPY it and use the copy.

To RENAME the backup file:

1. From the Opening Menu, press E for RENAME.
2. From within a document, press ^K E.
3. In answer to the prompt, type the name of the backup file and press RETURN.
4. In answer to the next prompt, type a new name that does not have the .BAK extension.

You can now edit the file as usual.

The CP/M renaming utility REN can also be used from the A > prompt.

To COPY the file:

1. From the Opening Menu press O for Copy.
2. From within a document, type ^K O .
3. In answer to the prompt, type the name of the backup file and press RETURN.
4. In answer to the next prompt, type in a new name, without the .BAK extension, for the copy

You can now edit the copy.

Changing Disks While in WordStar

Do not remove either the WordStar disk or the data disk from the machine while you are editing a file. WordStar needs access to both the program and the data disks during editing. If either disk is removed, you will eventually get an error that will erase the file you were editing (only the current editing session will be lost, not any copy of the document stored on disk).

You can safely change disks only when you are in the Opening Menu.

"BDOS Errors"

Occasionally the phrase "BDOS ERROR" may appear on the screen; it may happen when you try to log a disk, or save a file, or when WordStar goes to the disk to pick up additional data.

"BDOS error" means the computer is having trouble reading or writing information on the disk. The message may mean there is a flaw on the disk, which is difficult to recover from (a reason to keep recent copies of all disks). The message will also appear if the disk drive is left open, or if no disk is in the drive; that problem is usually solved by putting in the disk or closing the drive. The problem might also be transient: if you were changing the logged disk, for example, the error message may go away if you repeat the command. If the computer just can't read the disk, though, you will have to turn to your backup copies.

Before Editing

Advanced Level

Naming Files
Converting Files
Saving Crashed Files
Non-Document Files
Overlays

More on Naming Files

WordStar creates CP/M files; naming conventions and limitations for WordStar are the same as for any CP/M file.

"Non-Document" Files

N *Open A Non-Document File*

A Non-Document file is one intended for use outside of WordStar. It might be text to be inserted into a BASIC computer program, or data for a MailMerge file or a database listing.

A Document file is not suitable for these purposes because of the special features WordStar offers in Document mode. WordStar alters the standard computer character codes in order to keep track of its special editing functions such as "soft" hyphens, RETURNs, and spaces. WordStar alters the "high order bit" of the ASCII code, which would otherwise be left at zero.

The altered ASCII characters cannot be read by some programs like BASIC. The Non-Document file option solves this problem by eliminating the special WordStar features that alter the character codes, thus allowing input through WordStar for use in non-WordStar-compatible programs.

Non-Document files are created or edited in the same way as Document files: press N while in the Opening Menu; type in the name of the file; press RETURN. Non-Document mode can be used to edit files created using Document mode, and vice versa.

Non-Document mode has the following defaults. These settings can be changed from the On-Screen Layout menu.

Wordwrap is OFF in Non-Document mode. Therefore you must use the RETURN key as if using a conventional typewriter.

The Top Line of the on-screen menu does not display page number or line in Non-Document mode. Instead, the file character count (FC) and the number of file lines (FL) is shown. See *Page Break Display* in the *On-Screen Layout* section for more details.

TAB changes from 5-space Variable Tabs as in Document files, to Fixed Tabs every 8 spaces. In Non-Document files, the TAB key is a single character, which can be deleted or edited like a RETURN or any letter. Fixed tabs are permanent and cannot be altered, deleted, or added. See the *Fixed Tabs* paragraph of the *On-Screen Layout* section.

Other defaults found in Non-Document mode:

Page Breaks not shown
Reformatting unavailable
Hyphen Help is off
Right Justification is off
The Ruler Line normally at the top of the page is not displayed
Dot commands are ignored

Converting Document to Non-Document Files

It is possible to convert Document files to Non-Document type using the PIP command as follows (also see *Volume 4—Operating Systems—CP/M Plus.)*

P I P SPACE BAR A : = B : filename [Z]
RETURN

where A: is the destination drive, B: the source drive, and [Z] the special PIP function that "zeroes out" all the high-order bits, thus taking away the codes WordStar had added. The resulting file can be used as the basis of a BASIC program, for example. However, it cannot easily be re-edited by WordStar since the [Z] function also converts all the "soft" RETURNS to "hard" RETURNs and they cannot be readily converted back again.

Converting Non-Document to Document Files

You can always use Non-Document mode on a file created under Document mode; and you can Document edit a file created under Non-Document mode. The mode only affects the WordStar features available.

Files created in Non-Document mode have all hard RETURNs, which cannot be formatted in the Document mode. To convert a Non-Document file so it can be easily used in Document mode, you must delete all the extra hard RETURNs and replace them with soft RETURNs. Details on hard and soft RETURNs and on the other commands discussed here are given elsewhere; they are summarized here for convenience.

Here is how to convert Non-Document to Document files:

Use Document mode to edit the file. To replace all hard RETURNs with soft ones, use the Find and Replace (see **Find and Replace**):

1. Type the command ^Q A.
2. In response to the prompt "Find?" enter ^N RETURN.
3. In response to the prompt "Replace with?" enter a space with the spacebar, then RETURN.

4. In response to the prompt "Options?" enter N RETURN.
5. The cursor jumps to the first hard RETURN and replaces it with a space, causing the first two text lines to join together as one long single line (it may spill over to the next line on the screen).
6. Use the command ^L to repeat this Find and Replace selection so that the next hard RETURN is replaced by a space. Repeat this ^L command all the way through the document, skipping over the RETURNs that separate paragraphs.
7. The result is a series of one-line paragraphs (with overflow). Now return to the beginning of the document and use the ^B reformat command to reformat each paragraph (see the *Reformatting* section).

Your document is now in recognizable paragraphs with soft RETURNs at the line endings. Normal Document commands can now be used.

WordStar Overlay Files

Although CP/M Plus permits changing disks without causing a BDOS error, WordStar uses both disks for overflow files during editing.

WordStar is too large to be kept in the computer's memory all at once. Less frequently used commands are stored in "overlay files" on the WordStar disk and retrieved as needed. Similarly, overflow from the document being edited is stored on the data disk, permitting document files theoretically as long as the disk capacity (there are other practical limits, however). This is why the WordStar disk drive sometimes starts up during editing, causing a slight delay in command response time.

If the WordStar disk or the data disk is removed, an "overlay error" ("Error E38") will occur the next time a WordStar overlay file or a data overflow file is needed by the program. This error is fatal to the file being edited, causing your current editing session to be lost (only the active file is lost, not whatever is stored on disk).

Saving Crashed Files

For those familiar with the CP/M program DDT, it is possible to salvage WordStar text following a program failure. The CP/M 3 version of DDT is called SID and is located on your CP/M Plus disk. WordStar 3.3 stores text beginning at memory location 846EH. Text can be recovered even following a RESET as long as the power supply has been maintained and no new input has written over the memory location. Consult your *Volume 4—Operating Systems— CP/M Plus* documentation for details on the SID program.

Inserting and Cursor Movement

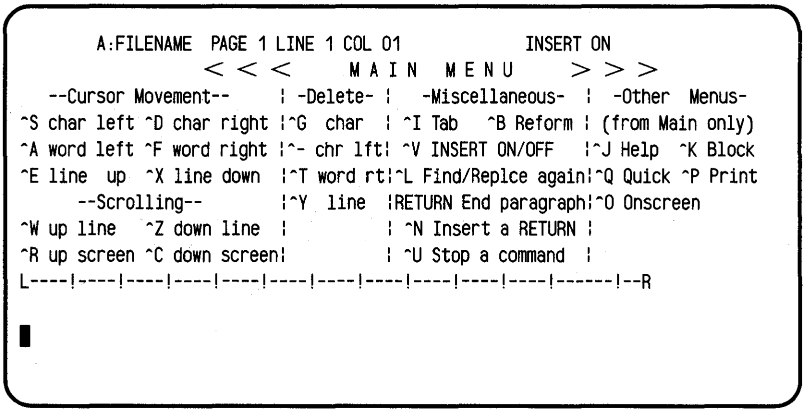
Basic Level

Cursor	*Typing*
Spacebar	*Wordwrap*
Special Keys	*Moving the Cursor*
Substitute Arrow Keys	*Move By Word*
Move By Screen	

The Cursor

The **CURSOR** is the small block that shows your position on screen. Anything you type appears on the screen at the cursor position. Most commands you issue also take effect at the cursor position.

To practice cursor movement, first open a Document file. When a new file is opened, a screen similar to the following will appear:



```
        A:FILENAME  PAGE 1 LINE 1 COL 01             INSERT ON
                   <<<      M A I N   M E N U      >>>
   --Cursor Movement--     ¦ -Delete- ¦  -Miscellaneous-  ¦ -Other  Menus-
^S char left ^D char right ¦^G  char  ¦ ^I Tab   ^B Reform ¦ (from Main only)
^A word left ^F word right ¦^- chr lft¦ ^V INSERT ON/OFF   ¦^J Help  ^K Block
^E line  up  ^X line down  ¦^T word rt¦^L Find/Replce again¦^Q Quick ^P Print
      --Scrolling--        ¦^Y  line  ¦RETURN End paragraph¦^O Onscreen
^W up line   ^Z down line  ¦          ¦ ^N Insert a RETURN ¦
^R up screen ^C down screen¦          ¦ ^U Stop a command  ¦
L----!----!----!----!----!----!----!----!----!----!----!------!--R

█
```

SCREEN 4. Main Menu

The cursor is shown in the left corner of the screen. If there is no text on the screen, you cannot move the cursor (use the **Arrow Keys** to try). You must type something on the screen for the cursor to move on.

Typing Text

You can write text on screen by typing at the keyboard, in the same way as typing at a typewriter. Practice typing and see how the words appear on the screen, the cursor moving across the screen as you type.

Type several paragraphs of text on which you can demonstrate the commands. Don't worry about typing errors you may make: we will show you how to correct them in a moment. First, here is an important tip: as you reach the end of the first line of typing, DO NOT press the RETURN key. Instead, just keep typing.

As you pass the right margin, the next word you type jumps down to the beginning of the next line, just as if you had used the RETURN key. This is called Wordwrap, a great convenience because you no longer need to watch the right margin as you type. The only time you need to press RETURN is to end a paragraph.

Down the right side of the screen is the FLAG COLUMN, where characters mark the kind of material on screen. In the flag column dots indicate empty lines; a left caret mark indicates a hard return; a blank indicates a line ending in a soft return; a MailMerge dot command is shown by an "m"; and an incomplete or erroneous dot command by a question mark.

Three Quick Tips for Making Corrections

You may wish to make corrections or additions to your text as you type it in. WordStar's many editing commands are described in more detail in later sections. For now, though, here are three quick editing tips to let you make some basic corrections.

1. *To move the cursor to any place in the text, use the Arrow Keys.*
2. *To add a letter, word or space, move the cursor to the exact spot where the item must be added. Then start typing. What you type will appear in the text, and whatever follows it will be pushed out of the way. Spaces and RETURNs can be added just as if they were letters.*
3. *To delete a character, or space, or RETURN, hold down the* CTRL *key and press* G*. This erases the letter at the cursor.)*

Insert

You will see the words **INSERT ON** at the upper right of your screen. As long as INSERT is ON, what you type is inserted at the cursor position.

INSERT can be turned OFF. On the Main Menu under "Miscellaneous" there is the reminder "^V INSERT ON/OFF." Entering ^V (i.e., hold down the CTRL key and press V) causes the INSERT ON message to disappear from the top line of the screen. With INSERT OFF, the text you type will write over existing text on the screen. (Text can always be added to the end of existing material, whether INSERT is ON or OFF.)

INSERT is a TOGGLE COMMAND: it may be turned back on by entering the ^V command again. There are many such toggle commands in WordStar.

Spacebar

If you use the Arrow Keys to move the cursor around the screen, you will find that the cursor can be moved any place on screen where there is text or spaces you've typed in with the spacebar. But the cursor cannot move to parts of the screen where nothing has yet been written. To move into an "empty" area of the screen, use the spacebar.

The **SPACEBAR** adds spaces. If you use it within text, the text will move aside just as if you were typing in words. This is one way to move text around on the screen.

Wordwrap

When you're typing and the cursor reaches the right margin, the next word you type will jump down to the beginning of the next line, as if you had used the RETURN key. This is called Wordwrap, and the RETURN that WordStar inserted for you is called a **soft RETURN** — ones you put in are called **hard RETURNs**.

WordStar is able to add or delete soft RETURNs during editing; hard RETURNs cannot be deleted by WordStar: you have to delete them yourself.

For this reason, you should not use the RETURN key when you reach the right margin. Instead, only use the RETURN key to signal the end of a paragraph or to put spaces between paragraphs. If you put RETURNs into the middle of a paragraph, you will lose some of WordStar's most useful editing features.

Wordwrap also automatically **right justifies** your text on screen. This means the right side of the text lines up evenly, the same way it does in a book or newspaper.

WordStar justifies the right margin by adding extra spaces between words on some lines. These are **soft spaces**: WordStar put them there, and WordStar will remove them if they are no longer needed, or add more when necessary. Spaces you put in with the spacebar are hard spaces and WordStar won't delete them. You can delete both hard and soft spaces.

Special Keys

Use the character and number keys on the keyboard to enter those characters or numbers onto the screen at the cursor position. Typing at the Osborne Executive is like typing at a typewriter except that what you type appears on a screen instead of on paper.

TAB

The TAB key moves the cursor to the next tab setting, in the same way as a typewriter. Tabs are automatically set every five spaces when WordStar is started. The TAB is a repeating key.

Press the TAB key several times, typing in text between Tabs. The TAB actually adds five hard spaces to the line. To delete the indentation created by a Tab, delete each of the five spaces.

RETURN

The RETURN key is the same as on a typewriter.

There is one important exception, however: Wordwrap. Because of Wordwrap, you should only use the RETURN key at the end of a paragraph.

SHIFT

As on a typewriter, this key produces capital letters, or the character shown on the upper half of a number or symbol key. (Note: you can **redefine** exactly what each key stands for on the keyboard with the CHARGEN feature of the Osborne Executive. This gives you access to special symbols and characters from other languages. CHARGEN is explained in *Volume 4—Operating Systems—CP/M Plus.*)

CAPS LOCK

When pushed down, this key causes subsequent *letters only* to appear in capitals. Press the CAPS LOCK again to release. Numbers and symbols are NOT affected by the CAPS LOCK: you still have to use the SHIFT key on those.

CTRL

In Osborne documentation and in the WordStar menus, the symbol for the CTRL key is ^. The CTRL key is used together with another key to form a command that WordStar will follow. For example, ^G means "hold down the CTRL key and press the G."

The CTRL key is like a shift key and is used the same way. When it is held down, the keys you press issue commands to WordStar instead of letters.

In the Opening Menu, you are only issuing commands, not typing in text; so WordStar can be addressed directly with single-letter commands and no CTRL key.

But once you enter a document to edit or write, you must be able to distinguish between keystrokes you intend as text to appear on screen, and keystrokes you intend as commands.

You use the CTRL key to do this, signaling to WordStar that the next key pressed is part of a command rather than text.

Moving the Cursor

△▽◁ ▷ *Arrow Keys*

The four **arrow keys** on the keyboard cause the cursor to move one space or character right or left, up or down. When the cursor reaches the end of the existing text on a line or in a document, it stops.

Like all the character keys on the keyboard and most of the command keys, the arrow keys are repeating keys. Hold them down and the cursor will move rapidly across the screen.

Substitute Arrow Keys

The following commands move the cursor in exactly the same way as the arrow keys: **^S**; **^D**; **^E**; **^X**. They form a compass on the keyboard just like the arrow keys:

(Use with CTRL key:)

Up a Line
E
Left Character S D Right Character
X
Down a Line

These commands are often used by typists who prefer to have the cursor commands available in one hand.

You will find these commands listed under "Cursor Movement" in the Main Menu. They were originally devised for computers lacking arrow keys, but many WordStar users prefer them over the arrow keys.

Try using these commands as a substitute for the arrow keys to determine which approach is most suitable to your typing style.

Moving the Cursor By Word

There are several ways to move through the text faster than simply using the arrow or substitute arrow keys. These commands are used while you are editing a document.

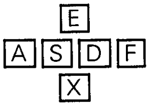
^F ***Word Right***

You can skip over one word to the right in the text. If you are at the end of a line, the command moves the cursor to the beginning of the next line. This is a repeat key.

^A ***Word Left***

Moves the cursor left one word (or to the next previous space). At the beginning of a line, it moves the cursor to the end of the previous line. This is also a repeat key.

These commands are extensions of the substitute arrow keys, as seen in this diagram:



Moving By Screen

Not all of your document can appear on the screen at once. As you fill the screen with text, the top of your document disappears under the Main Menu. Later you can return to view it, and the bottom part of your document is hidden from view down "below" the bottom of the screen.

The computer views your document like a long scroll, with the screen giving a view of whatever part of the document you wish. When you are in the middle of a long document, the text you've passed is "above" you in the document, the rest "below" you. To see the next section of the text or the previous section, use these commands:

^R *Up a Screen*

Use this command to display the previous screen of text (with two lines of overlap). You go up a screenful to see the 20 lines "above" your present position.

^C *Down a Screen*

Use this command to display the next following screen of text (if any), taking you down a screenful to see the 20 lines "below" your present position.

Inserting and Cursor Movement

Intermediate Level

Moving By Line
"Quick" Moving
Two-Letter Commands

Inserting

There are two additional insertion commands:

^N *or* ^M *Inserting a RETURN*

The first of these inserts a RETURN at the position immediately to the right of the cursor. The text is pushed down and away from the cursor, while the cursor remains at its original position in the text.

The main reason to use ^N instead of RETURN is that it pushes the text down (as RETURN would do) but leaves the cursor behind. (RETURN of course takes the cursor with it.) This is often used to create some visual space for editing and inserting.

^M is an alternative to the RETURN key and is identical to it in function in all ways. Like RETURN, the cursor moves WITH the text that is moved down by the ^M command.

^I *Substitute TAB Command*

This is a substitute TAB command, ^I, shown in the Main Menu under "Miscellaneous." It was devised originally for keyboards lacking a TAB key and works the same way as the TAB key.

Moving By Line

^W *Up a Line*

This command is used to display the line above the top line on the screen; it moves the text on the screen down one line to make room. This is a repeat key.

^Z *Down a Line*

This command displays the line below the last line on the screen; the text on the screen moves up one line. This is a repeat key.

Two-Letter Commands

Many WordStar commands consist of two parts: the control-key/letter-key combination you've seen until now; plus an additional key. To use these commands, hold down the CTRL key and press the first letter, as usual. Release both keys and press the second key.

For example: ^Q E means "hold down the control key and press Q, release both keys, then press E." In other words:

^Q then E

The reason for this two-part command is that the "^Q" command is actually a group of commands under the "Q" prefix. When you use ^Q, you are telling WordStar that you want one of the ^Q group. The letter E is the specific one of the ^Q group you want.

Some two-part commands used in a document are similar to commands used in the Opening Menu. The extra control commands signal the beginning of a command. For example, O = Copy a File in the Opening Menu corresponds to ^KO in a document; and E = Rename a File is the Opening Menu version of the ^KE command used in a document.

"Quick" Moving

The ^Q group of commands is called the "quick" group because it includes cursor commands to jump from place to place in your text more quickly than by moving letter by letter or line by line. These commands are all used during file editing (i.e., while you are in a document).

^Q E *Top of Page Jump*

Use this two-part command to jump the cursor to the top left corner of the current screen.

^Q X *Bottom of Page Jump*

This two-letter command jumps the cursor to the bottom right corner of the screen.

^Q R *Beginning of File Jump*

This command jumps the cursor to the beginning of the current file.

Warning!

If you are moving backwards *a long way in the file—over four pages or so—it is advisable to move to the beginning of the file by using* ^[K][S] *("temporary save") instead of* ^[Q][R].
The text will be saved to the disk, and you will be returned to the beginning of the document — the result is the same, but it avoids some errors that can arise from jumping too far back and forth in the file. See File Handling *for more information on the ^KS command.*

^[Q][C] *End of File Jump*

This command jumps the cursor to the end of the file.

^[Q][S] *Left Side Jump*

Moves cursor to column 1 (the first column on the left) of the line it's on.

^[Q][D] *Right Side Jump*

Moves cursor to the last character of the line it's on.

Inserting and Cursor Movement

Advanced Level

Continuous Scroll

Continuous Scroll

^Q W *Continuous Scroll Up*

^Q Z *Continuous Scroll Down*

These commands will initiate a continuous, slow, line-by- line scrolling through the document, up or down, until cancelled by the Spacebar, or until the cursor reaches the beginning or end of the file. This is a convenient way to view the entire document before printout.

Press the Spacebar to stop the scroll.

Deleting

Basic Level

Single Character
Word Right
Line Right
Character Left
Entire Line
Line Left

Deleting Text

You can delete or erase text from your document with a variety of WordStar delete commands. Some of them erase one character at a time, some a word, and some delete a whole section of text at once. First we'll describe basic delete commands; the more complex ones appear in the intermediate and advanced levels.

When these delete commands are used to erase something from the text, the remaining text closes up to fill in the space created by the deletion.

^G ***Delete Character (At Cursor Position)***

^G is the Delete Character command. The character at the cursor position is erased. Whatever follows the deleted character moves to fill in the gap.

Move the cursor to a character in the middle of a line of text. Use the ^G command once. The remainder of the line is pulled in to fill the gap.

Move the cursor to a space between two words. Use the ^G command to erase the space as if it were a character.

If you move the cursor to the space at the end of a line and use the ^G command several times, you will eventually erase the

RETURN at the end of the line. The next line will move up to join your line. (See **Reformatting** later in this section to return the paragraph to its original condition.)

Like many commands, ^G is a Repeating command: if you hold down the ^G combination, it will rapidly erase character after character, the text to the right being reeled in and deleted. Watch the screen when you delete to make sure you don't erase more than you intended.

^- *or* ^◁ *Delete Character Left*

(The dash key is next to the equals sign on the upper row of the keyboard.)

These two commands delete to the left of the cursor. They are identical in their operation. When you use either command, the cursor moves left, erasing characters as it goes. Commonly used to "back up" and erase while inputting, ^- and ^◁ are also repeating commands.

Note:

> *^- and ^◁ both correspond to the DELETE (or DEL) key referred to in some computer manuals. Using these commands does the same thing as pressing the DEL key on computers that have one.*

^T *Delete Word Right*

This command deletes the word at the cursor position, erasing from the cursor to the right.

If the cursor is on the first letter of a word, ^T erases the entire word, and spaces following it. If the cursor is in the middle of a word, ^T erases the right half of the word.

Example: EXEC<u>U</u>TIVE Computer ↑ cursor   EXEC_Computer ↑ cursor

^T pulls the word "Computer" up to the cursor position.

If the cursor starts on a space, all spaces on that line up to the next word are erased.

If the word being deleted is at the end of the line, the RETURN following it is not erased. Repeat the ^T command to erase the RETURN.

^T is also a repeating command.

^Y *Delete Line (All)*

This erases the ENTIRE line the cursor is in. It doesn't matter where in the line the cursor is located, the whole line is deleted.

Warning:

^Y is a repeating command. There may be a slight pause while WordStar activates this command. ***Don't hold down the ^Y command while waiting for it to take effect, because that may delete several extra lines by mistake.*** *Simply issue the command and release the keys: WordStar will register and carry out the command for you.*

A Quick Look at Reformatting

The **On-Screen Layout** section of this manual has details about reformatting. But in the course of adding and deleting material from your document as you practice these commands, you will find your paragraphs growing ragged and hard to read. So here is a quick four-step lesson in "reformatting," which smooths out those jumbled paragraphs in just a stroke.

Here's how to reformat:

1. Put the cursor anywhere on the first line of the paragraph that needs its line endings straightened out.
2. Enter the command ^B. This is the Reformat command.
3. WordStar will go through the paragraph, fill in short lines and break off long lines until the right margin of your paragraph is even again.
4. If WordStar can't fit a word into the line without going past the right margin, it asks if you want to hyphenate the word. For now, just **bypass** this decision by repeating the ^B command any time you are asked to hyphenate. This will make WordStar continue through the paragraph.

Deleting

Intermediate Level

Line Right	*Line Left*

More Ways To Delete

^QY ***Delete Line to the Right***

This is a two-letter command: ^Q, followed by Y.

Place the cursor in the middle of a line of text. Enter the ^QYcommand: the rest of the line is deleted, from the cursor position up to the end of the line (not counting the RETURN). This is a repeating command.

^Q ^- *or* ^Q ^ ◁ *Delete Line to the Left*

This command deletes a line of text from the cursor position to the left margin.

This is the only two-key command in which it is necessary to hold down the CTRL key for **both** letters in the command (both the Q and the dash, or both the Q and the Left Arrow).

Deleting

Advanced Level

Deleting From Within File

Advanced Deleting

^K J *Delete File (While Editing Another File)*

With this command you can delete any file on either disk while editing another file. After you enter ^KJ, WordStar asks you for the name of the file to delete. If the file is not on the logged drive, precede the file name with the drive letter (A: or B:). **You cannot delete the file you are editing.**

This function is very useful for recovering from a DISK FULL message during a save. Deleting a backup file or another unneeded file will create the additional space needed on the disk. More details are given in the **File Handling** section.

Block Commands

Basic Level

Begin Block
Hide Block
Copy Block
End Block
Move Block
Delete Block

Marking Blocks

You can mark off a section of text, called a **block**, and manipulate it: copy the block, move it to another place in the text, write it as a separate file elsewhere on the disk, or delete it. Blocks are part of the ^K group of two-letter commands.

To "block" off text, mark the beginning and the end of the block using these codes:

^K B ***Begin Block***

First move the cursor to the beginning of the section of text you want blocked off. Then enter the command ^K B: the symbol <B> appears on the screen to show where the block begins.

^K K ***End Block***

Now move the cursor to the end of the section being blocked off. Enter the command ^K K: the symbol <K> appears on the screen to show where the block ends.

If you've marked the beginning of the block (^KB), setting the end marker will cause all the text between the <B> (beginning) and the <K> (end) to dim on screen. The dim or half-intensity text is the block. (The half-intensity is on the screen only: it won't affect printout.) The Begin and End markers disappear during this display.

Once you've marked off the block, you can manipulate it with the commands shown below.

If you only mark one end of the block, or put the end marker before the beginning marker, you'll get an error message when you try to use the block. Press ESC and correct the mistake.

There is only one block active at a time. You can reset either or both markers to new locations at any time, shrinking, enlarging or moving the block as a result.

Moving the block moves the block markers with it; deleting the block leaves the markers behind, side by side but turned off (hidden from view). If you start to set the blocks and then change your mind, you can get rid of the ^B display by adding a ^KK anywhere, then entering ^KH to turn it off (see below).

^KH *Turn Off (Hide) Block*

To turn OFF the half-intensity block display, or to turn it back on, use this command. It's a **toggle,** which means using ^KH will switch the block display back and forth between ON and OFF.

Only the half-intensity display is turned off: the block markers themselves are still there, but hidden from view.

Blocks can only be used when ON (that is, when the block is at half-intensity). Toggle ^KH back on (i.e., repeat the ^KH command) to redisplay the half-intensity block markers or the block itself.

It doesn't matter where you are in the document; the block can be used, turned on and turned off (you need not be near the block to use it).

Block markers are deleted when the file is saved to disk.

Using Blocks

Note:

> *When moving or deleting blocks, make sure you don't split up boldfacing or underlining command pairs.*

^K V ***Move Block***

This command moves the block to a location of your choice in the file. Existing text at the new location is pushed to the right just as if you had manually typed the material. The original block is deleted from its old location.

1. ^K B to mark the beginning of the block.
2. ^K K to mark the end of the block.
3. Move the cursor to the new location.
4. ^K V to move the block.
5. ^K H to turn off the block display.

^K C ***Copy Block***

This command places a copy of the block at the location of your choice in the file. It operates the same as the Move Block command except that the original of the block remains in its starting location. The copy of the block, now at the cursor location, is now the block and is displayed at half-intensity. It can be altered, recopied, moved, or turned off.

1. ^K B to mark the beginning of the block.
2. ^K K to mark the end of the block.
3. Move the cursor to the new location.
4. ^K C to copy the block.

Quick Tip:

REPEAT COPYING

To repeat a line or paragraph throughout a text, block off the first occurrence, then copy it each time you reach a new location where it's needed.

^K Y *Delete Block*

This command deletes the block. The space is filled by whatever text follows.

1. ^K B to mark the beginning of the block.
2. ^K K to mark the end of the block.
3. ^K Y to delete the block.

Quick Tip:

CHANGING YOUR MIND

If you issue the ^K command, you can cancel it simply by pressing the ESC key. Once you have entered the second key of the ^K command, and the process you ordered begins, the computer is usually too fast to be stopped by the usual Interrupt command, ^U.

Block Commands

Intermediate Level

Block Size

The size of the block is limited by available memory. A block can be no larger than about six single-spaced pages of text.

Marking Column Blocks

^K N *Column Block ON/OFF*

WordStar is able to mark off and move blocks of text in **columns**. A column is a square or rectangular area you define within the text, and move, copy, delete, or write to disk like an ordinary block.

Here's an example of two columns:

One Column	*Another Column*
This rectangle of the text is one of the columns. It can be marked off by block commands and used just like any other block, as you'll see below.	This is the other column, which you can see is next to the first column. With the Column Block command in WordStar, you can block off this column separately from the one on the left.

The Column Block mode is turned on and off with the **toggle command ^KN**. It's OFF unless you turn it ON. To check the status of the command, enter ^K and wait for the ^K menu to appear on the screen; look for "N Column now ON" or "OFF".

When you turn the Column Block mode ON, block commands now function in Block mode as described below.

^K B ***Begin Column Block***

As usual, you begin a column block by marking it with a ^KB. In this case, however, you are marking off the **upper left corner** of the column.

^K K ***End Column Block***

Here you mark the **lower right corner** of the block. As you do, the block itself goes to half-intensity. (You can then adjust your block markers if necessary.)

Using Column Blocks

Having marked the column block, place the cursor at what will be the **upper left corner** of the new column block location, and use one of the Block operations: ^K C (Copy); ^K V (Move); ^K Y (Delete).

To move the cursor into an area of the file where you haven't written anything yet, you may have to use the spacebar. The column will fill in the rest of the area, adding spaces as needed to keep the column intact in its original form.

If you move your column into the midst of existing text, the existing text will be bisected to admit the new column. In the same way, if you take a column out of the middle of existing text, the text that remains behind will close up to fill the gap.

If you move two columns next to one another as we did in the sample above, WordStar considers the two columns to be one big paragraph after that, even though visually they are two. The reformat command and other editing functions will treat the two blocks as one paragraph separated by spaces.

Column Block commands are particularly useful for moving columns of numbers in reports and for constructing multi-column newsletters.

The size limitations on column blocks are somewhat more stringent than for other kinds of blocks because the computer must include the material around the column in its calculations. A "Block too long" error message may occur more often. Simply subdivide the column and move it in smaller pieces. Maximum column width in any event is 250 characters.

Block Commands

Advanced Level

Cursor Block Moves

The following commands jump the cursor to block locations in your text.

^Q B ***Move To Beginning of Block***

Use this command to jump to the Begin Block <B> marker, whether currently displayed or not.

^Q K ***Move To End of Block***

This command jumps the cursor to the End Block <K> marker.

These commands turn the display ON if it is OFF.

They will also jump the cursor to the <B><K> markers which are left behind side by side after a Delete Block command.

^Q V ***Move To Previous Block Location***

This command moves the cursor to where it was when you first marked the block (^KB).

^K0 - 9 *Set Place Markers*

These commands are used to mark off important places in your text during editing.

To set a Place Marker, positon the cursor at the place you wish to mark. Enter the command ^K, followed by a number between 0 and 9; a numbered marker appears in the text at the cursor. You can then use the ^Q commands below to jump to the location of that Place Marker.

The Place Marker remains displayed until you set it in another place in the text, or until you save the text to the disk. The marker is not saved. To turn off a marker, put the cursor beside the marker and repeat the Set command using the same number.

^Q0 - 9 *Jump To Place Marker*

This command jumps the cursor to the Place Marker you set using the command above. Enter ^Q followed by a number from 0 to 9.

The ^Q0-9 command is faster than most other ways of moving through your text. The maximum length of cursor movement **backwards** is about 10 pages. More than this may introduce errors into your text.

Writing To Disk

^KW *Write Block To Disk File*

Block Mode: This command creates a new file on the disk containing a copy of the block you've marked.

This command is issued while you are editing a file. Block off the section of text you wish and enter the command ^KW. WordStar asks you for a filename to use for the block. Enter a filename following the usual rules for WordStar filenames.

Press RETURN, and the block is saved on the logged disk under that filename.

You cannot transfer a block into an existing file on the disk. Use the Read a File command described below to accomplish the same thing in another way.

You can store the block on the other (nonlogged) disk by preceding the filename with the disk drive letter and a colon. For example, to file SAMPLE.ONE on the nonlogged drive A, answer the filename prompt with

A : S A M P L E . O N E.

This ability to write to either disk is especially useful for recovering from a DISK FULL error message, when you need to break up your document into smaller files for storage.

The original of the block is still in the file you are editing and can be copied, deleted, moved, or rewritten under another name.

The Interrupt Command, ^U, cancels the Block Write commands at any time before you press the final final RETURN key.

Writing Out Column Blocks: WordStar won't write to disk or read in a file while Column Block mode is turned ON.

To write a column block to disk, therefore, follow this procedure:

1. Mark off the Column Block and move or copy it to the end of the file.
2. Enter ^K N to turn OFF the Column Block mode.
3. Use the ^K W command to write the block to disk.

Inserting a File From Disk

^KR *Read Another File Into Open File*

This command allows you to merge files together.

1. Open a file for editing.
2. Move the cursor to where you want to insert the second file.
3. Type the ^KR command. Answer the prompt with the name of the file you want.

A copy of the file is inserted in the document you are editing, beginning at the cursor and moving all following text out of the way. The original file still remains on the disk.

A block of text can be moved from one existing file to another one in two steps. First mark the block off in the first file and write it to disk. Then edit the second file and read in the block from disk.

When inserting a file, use ^KS to backup the resulting file. This will protect against loss of data as the open file grows larger. This is particularly advised for insertions reaching about ten pages in length. It is further advisable to break up files larger than ten pages into several smaller ones before inserting them into an open file; use ^KS to save the growing open file between insertion commands.

WordStar won't read in a file while Column Block mode is turned ON.

Related Commands

^K F *View File Directory*

This command turns on the File Directory listing of the logged drive while you are editing a document. This command is a toggle: enter it a second time to turn the directory off.

^K L *Change Logged Drive*

To see the other disk drive directory while editing a file, first use this command to change the logged disk drive. The File Directory will then show names of files on the other drive.

When you change the logged drive while editing a document, blocks written out to file (^KW) will go to the newly logged drive unless you specify otherwise. However, the document you are editing will still be saved on the disk it's presently on.

If an open file is entirely in memory (has no overflow on disk), it may be possible in an emergency to remove the logged data disk and log in a new one, then save the open file on the new disk; however, this is risky and won't work consistently.

^K J *Delete A File From Disk*

This command lets you delete a file that is on the disk while editing another file. It is the in-file equivalent of the Y command you see in the Opening Menu. It can be used to delete a temporary file, or to create extra room on the disk when in danger of having a full disk.

Type the ^KJ command and answer the prompt that asks for the name of the file to delete. Put the disk drive letter and colon before the filename if the file is not on the logged drive. You cannot delete the file you are editing (you can delete its existing backup file, however). To cancel this command, use ^U before pressing the final RETURN.

Find and Replace

Basic Level

Find	*Find & Replace*
Repeat Find & Replace	

The "Find" command in WordStar quickly searches through the text to find any character, word, phrase or symbol you want. It jumps the cursor to a specific part of your document.

The "Find & Replace" command searches for the requested item, then replaces it with a character, word, phrase or symbol you choose.

Use Find & Replace, for example, to replace one name in a document with another name; or to insert a long standard phrase (like "Atchison, Topeka & Sante Fe") without typing it repeatedly.

You can Find and Replace any item or keyboard character except certain command characters. WordStar will Find and Replace items up to 30 characters long.

Enter the ^Q part of any Find or Find & Replace command, then pause: the **Quick Menu** appears in the menu area of the screen. This menu summarizes the commands explained in this Find and Replace section.

Find

^QF *Find an Item in the Text*

To use, type the command ^QF.

This prompt appears in the upper left corner of the screen:

```
        FIND?
```

Type the character(s) you're looking for. You then have two more choices:

Press ESC to bypass the Options list.

Or press RETURN to get the following prompt:

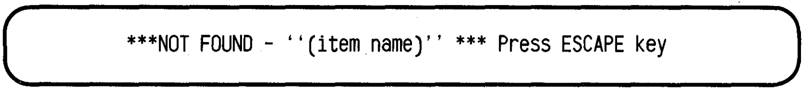
```
        OPTIONS? (? for Help)
```

The options include searching for whole words only, ignoring capitalization, and counting down a specific number of examples. The options are explained in the Intermediate Level below. You can also get an on-screen list of them by pressing **?** at this point.

If you wish to bypass the Options, press RETURN again.

After you've specified the item to Find or selected your options, **Find** searches the file for the first occurrence of the item specified. "First" means first occurrence after the cursor position when you issued the command. You can use the Find command from the beginning of the file, or from some point within the file: the search will proceed forward from that point.

Find will go all the way to the end of the file looking for the item specified. If it doesn't find the item, you'll see this message:



```
***NOT FOUND - ''(item name)'' *** Press ESCAPE key
```

The item you asked for is repeated ("item name") so you can check that you got it right. Press the ESC key to leave the Find command, leaving the cursor at the end of the file.

Quick Tip:

CURSOR MOVEMENT

Enter ^Q P *to jump the cursor back to its position when you gave the Find command.*

The computer is very literal-minded; it looks for **exactly** *what you typed. For example: if you ask for "the" it won't catch "The" because of the capital T. (In a moment we'll describe the Option for ignoring capitalization.)*

When looking for an item you requested, **Find** *will also include soft spaces inserted between words by WordStar. That's how exacting* **Find** *can be. You'll quickly get a feel for Find's way of looking at things.*

To interrupt a search, use the Interrupt command ^U

Find & Replace

^Q A *Find & Replace*

Use this command to find an item and replace it with another. When you enter ^Q A, WordStar responds with the **Find** prompt:

```
FIND?
```

Type the item you want to find (up to 30 characters), then press RETURN. The next prompt is:

```
REPLACE  WITH?
```

Type the new item (up to 30 characters) and press RETURN. (Or press ESC to bypass the Options prompt.)

You have two choices again.

1. Press RETURN to display the "Options?" prompt. Then select one or more of the Options or press RETURN to skip them.
2. Or you can press ESC to bypass the Options prompt.

WordStar then proceeds to the first occurrence of the phrase you requested, and STOPS.

In the upper right corner of the screen you will see:

```
REPLACE?  (Y/N)
```

To confirm that you want to replace the item here, press Y for Yes. The phrase is immediately replaced.

Use the Reformat (^B) command to fix the line endings and the text's appearance.

If you DON'T want to replace the item at this location, press N for No. The **Find & Replace** command ends and the cursor remains at this point in the text.

To interrupt Find & Replace, use the Interrupt command ^U.

^L *Repeat Find/Repeat Find & Replace*

Using this command will immediately repeat the Find or Find & Replace command last entered. Simply enter the command ^L — the rest is automatic, and faster than re-entering the sequence.

This command works even after you save one file and open another, as long as you haven't restarted WordStar.

Quick Tip

USING FINE & REPLACE FOR DELETIONS

Find & Replace can be used to zip through a file and repeatedly DELETE as well as simply replace. Here's how to Delete using Find & Replace:

1. *Enter* ^Q A.
2. *Type in the phrase you want to delete.*
3. *In response to the REPLACE WITH? prompt, press* RETURN *— DON'T enter any characters, just the RETURN key. (That means "no replacement.")*
4. *In answer to the OPTIONS? prompt, press* RETURN.
5. *You will be asked to confirm the deletion when the item is found. Enter* Y *and the item is erased without replacement.*
6. *Using the* ^L *Repeat command, you can quickly skip down (by saying* **N** *to the "Replace? (Y/N)" prompt) to each place where you want to delete.*

Find and Replace

Intermediate Level

Options *Special Search Codes*

Options

When you enter ^QF or ^QA for a Find or a Find & Replace, WordStar asks "OPTIONS?" Here is a list of the options available.

When the Options are first called for, you can enter a question mark (?) to get an on-screen quick listing of these options.

W = Whole Words Only: Searches for an entire word; won't stop if your item is part of a longer word (if you enter "the", it won't stop at "anthem").

U = Ignore Case: Ignores upper and lower case in the item searched for. For example, specifying "the" will stop the cursor at "The" or "THe" or "tHe".

N = Replace Without Asking (Find & Replace only): Will replace an item without asking for confirmation (with "Replace (Y/N)" prompt).

G = Replace in Entire File (Find & Replace only): Goes through the whole file, replacing the item each time it occurs. Asks for confirmation each time. Combining **N** and **G** will replace the item throughout a document without stopping for confirmation.

B = Search Backwards: Starts at the cursor position and searches back toward the beginning of the file. (Otherwise Find and Find & Replace operate forward from the cursor position.)

0 - 9 = In Find, entering a number will move the cursor down to that occurrence of an item. For example, entering the number 5 moves the cursor to the 5th occurrence of the item. In Find & Replace, entering the number 5 will replace the first 5 occurrences (prompting for each one unless you also select "N").

Several of the options can be entered at once, in any order. For example, in Find & Replace, the options 5NGWU would search the entire file, replace without prompting the first five whole word examples of the requested Find item and ignoring upper and lower case distinctions.

Special Search Codes

The Find and Find & Replace commands can include special codes which increase their versatility. (As with all ^P commands, only the second part of the command actually appears on screen.)

^P ^A *Match Any Character*

Including this code in a searched-for item causes WordStar to look for a "wild card" character at that position in the item. For example, specifying NA^P^AION (which appears on the screen as NA^AION) will find the following: NATION, NASION, and NA.ION. Use this command to search for a variable character in the middle of a word.

^P ^S *Match Any Symbol Character*

This is similar to ^P^A except that it searches for spaces or punctuation marks, NOT letters or numbers. For example, NA^P^SION will find NA ION or NA*ION but not NATION.

^P ^O *Match Other Than the Next Character*

This code specifies a search for any character or symbol EXCEPT the one following the code. For example, NA^P^OTION means "find any **but** the T" and would find NASION and NABION but not NATION.

^P ^N *or* ^N *Match a RETURN*

This is a handy command for finding RETURNs in your text. To search for a RETURN, enter ^P ^N or just ^N in the FIND? portion of the Find or Find & Replace command. A ^N will appear on screen, and the command will look for the next soft or hard RETURN in the text.

Quick Tip

TURNING THE PAGE

You can "turn the page" of a document with the following:

```
FIND? ^N    OPTIONS? 55  RETURN.
```

***Find** will look for the 55th RETURN — since WordStar puts 55 lines on a page, the result is a cursor jump to the next page. Then enter ^L (the FIND REPEAT command) and you will be "leafing through" your file page by page instead of screen by screen.*

***Speed Up Find & Replace:** When you issue a Find & Replace with the options N (Replace Without Asking) and G (Replace in Entire File), the process can take some time since WordStar stops to display each occurrence of the item.*

You can speed up this process. When replacement begins, enter any nonprinting character — for example, press the ESC key, or ^X. WordStar will stop displaying each replacement as it is made. The operation is "behind the scenes" and nothing appears on screen until the end of the file is reached.

Be sure you've issued the Find & Replace command correctly because you won't see it in operation. Since nothing seems to happen after the command, you'll have to be patient if you're searching a particularly long file.

Find and Replace

Advanced Level

Finding WordStar Commands   "Boilerplate" Insertions

Finding WordStar Commands

Most WordStar commands that appear in the text can be the object of a Find or Find & Replace command. Dot commands, for example, can be searched for as if they were ordinary characters. Some commands are a bit more elusive, however, and may take some experimentation.

Most of the ^P group of print control commands can be located by the Find and Find & Replace command. There are five Print Control Commands that **cannot** be found or replaced in this way because they have special meanings for Find & Replace: ^A, ^S, ^O, ^N and ^U (see *Special Search Codes above).*

Here's how to Find and Replace commands in the ^P group. We use the Boldface command, ^PB (which displays in the text file as ^B), as an example.

To Find the Boldface marker, type the Find command ^Q F; enter ^P and then ^B (that's CTRL-B, not just B!). ^B appears on the prompt line.

To REPLACE an item with a print control character, enter the print control character in the Replace portion of the Find & Replace command. Be sure to use the CTRL key on BOTH the P and the B parts of the command.

This procedure works with all the usable ^P commands.

Quick Tip

"BOILERPLATE" INSERTIONS

You can use Find & Replace to insert a long, cumbersome phrase that appears over and over in a document. There are two ways to do it. We'll use "Atchison, Topeka & Santa Fe" as an example.

First Method:

1. *Enter a symbol (such as $ or @) in the text instead of the phrase.*
2. *Then go back and Find all occurrences of that symbol and Replace with "Atchison, Topeka & Santa Fe." Use* ^L *(Repeat) as often as needed.*

Second Method:

1. *Initiate Find & Replace before starting to type the document.*
2. *For Find? enter* RETURN *(meaning "nothing").*
3. *For Replace? enter* ***"Atchison, Topeka & Santa Fe."***
4. *For Options? choose* N *(Replace Without Asking).*
5. *As you write the document, when you're ready to insert the phrase:*

    - enter a **space**
    - *back the cursor into the space*
    - *and enter the Repeat Find & Replace command ^L.*

WordStar inserts "Atchison, Topeka & Santa Fe" at the cursor position, and you can continue composing your document.

On-Screen Layout

Basic Level

Reformatting
Centering
Hyphenation
Line Spacing

These commands affect the appearance of text on the screen.

Reformatting

^B ***Reformat Paragraph***

WordStar "right-justifies" as you type, lining up the right side of the text evenly along the right margin.

Editing changes can make these line endings uneven. ^B straightens them up again. Even with Right Justification turned OFF, reformatting will adjust line endings to fall within the right margin zone (see *Ragged Right (Justify)* later in this section).

To use Reformat, move the cursor to the first line of the paragraph to be reformed. The cursor can be at any point in the line.

Enter ^B. WordStar will reform the paragraph.

WordStar reformats by moving words between lines, and adding spaces between words to create an even right margin (except when Justify is OFF).

If a hyphenation decision is required, WordStar will ask (see below).

If you put the cursor on a line in mid-paragraph then enter ^B. WordStar will reformat the remainder of the paragraph starting at that line. This means you can reformat beginning at the point where changes were made to the paragraph.

Note:

Special Lettering commands such as Boldface and Underline (explained in a later section) insert special codes on the screen to show where special lettering will take place during printout. These codes will make the line endings of a paragraph **appear** *to be out of alignment, even after using the Reformat command. In fact, the line endings will be correct during printing since the special codes themselves are not printed. You can confirm proper paragraph line endings with ^OD, the "Command Display OFF" command, which will cause these special codes to disappear from the screen (they will still function during printout). Repeat the command to turn them back on.*

Hyphen Help

WordStar reforms the paragraph without using hyphens if possible. But when a word is too long to fit or to bring down to the next line, WordStar will stop reformatting at that word and display this message:

```
TO HYPHENATE, PRESS -. Before pressing -,
 you may move cursor:   ^S=cursor  left,
 ^D=cursor  right.   If  hyphenation  not
 desired, type ^B.
```

The cursor is on the word needing hyphenation, at the place in the word where the hyphen should be placed. A WordStar feature called Hyphen Help incorporates basic hyphenation rules to determine this.

You have choices:

1. Press the hyphen (dash) - key, and the hyphen appears at the cursor location. Reformatting of the paragraph then continues.
2. Move the cursor, using the arrow keys or the character movement commands ^S or ^D, to another part of the word. Press the hyphen (dash) key. The hyphen appears at this location, and reformatting of the paragraph continues.

   If the hyphen is beyond the right margin, however, the entire word is brought down to the next line and reformatting stops. To correct, erase the hyphen, move the cursor up a line, and enter ^B to begin reformatting again. This time, when asked to hyphenate, choose a location within the margins.
3. Press ^B to refuse hyphenation. The word is brought down to the next line and reformatting continues.

The hyphen created during reformatting is a "soft hyphen" and is displayed on screen at half-intensity. A soft hyphen prints out only if when it is the last character in a line.

If you edit and reformat a paragraph and move the hyphenated word to the middle of the line, the hyphen will remain visible on the screen at half-intensity. However, it will not print.

Should the hyphen be needed again in reformatting, WordStar will use it automatically.

A paragraph with unused soft hyphens will appear to have uneven line endings, but it will print out properly.

Centering

^[O][C] *Center the Line*

To center a line between the current margins, enter ^[O][C] while the cursor is on the line.

You may use this command when you reach the end of the line and before entering the RETURN; or you may move the cursor back to the line to center it.

An item can be re-centered due to changed margins or altered text.

The text to be centered can extend beyond both margins and will still be centered properly.

Centering ignores spaces preceding (leading) or following (trailing) the text, except special Non-Break Spaces created by the ^PO command.

Enter the ^O part of the centering command and pause: the ^O On-Screen Menu appears in the menu area of this screen. This menu summarizes the ^O two-part commands explained in this section.

If Special Lettering commands such as Boldface or Underline are used in a line, the codes they leave on screen may make the line appear to be centered wrong. However, it will print correctly. The ^OD, Command Display OFF, command will cause these special codes to disappear from the screen temporarily; repeat the command to turn them back on again.

Line Spacing

^O S 1 - 9

WordStar displays and prints text single-spaced unless instructed otherwise.

To change line spacing, use the ^OS command. WordStar asks you to select a spacing from 1 to 9. Enter the number desired.

Subsequent typing and reformatting will adjust to the requested line spacing.

Reformatting (^B) will not change the number of blank lines between paragraphs.

See also the dot command .LH in the *Print Controls* section for another way of creating temporary spacing variations during printout.

On-Screen Layout

Intermediate Level

Indent
Left Margin
Set Tabs
Decimal Tab
Hyphen Help
Justification
Right Margin
Clear Tabs

Paragraph Indent

^OG *Paragraph Indent*

This command is used to create indented paragraphs. It sets up a temporary left margin indented one tab setting to the right of the current left margin. This is called the Paragraph Indent or Temporary Left Margin command.

Enter the command ^OG and watch the rule line (the row of dashes and exclamation points at the top of the page). WordStar highlights the dashes from the left margin marker **L** to the first **!** to show the indented area. The next paragraph demonstrates the results of the command.

```
Type  in  text  to  see  the  effect  of  the  ^OG
Paragraph  Indent command.   As several lines  are
typed,  the left side of the paragraph is indented
to the first tab setting.
```

Pressing RETURN at the end of the paragraph ends the Indent command and the rule line highlighting.

You can enter a series of ^OG commands to indent the left margin as many tab settings as you wish. The highlights on the rule line will indicate the extent of the indent.

A paragraph already entered can be reformatted using the paragraph indent command: Move the cursor to the first character to be indented and enter the number of ^OG commands desired. Then enter ^B (Reformat). The paragraph will be reformatted with the new indented temporary left margin. Moving the cursor outside of the paragraph being reformatted will cancel the Paragraph Indent command.

Hanging Indents This is an example of a hanging indent: a paragraph whose first linen begins at the left margin but whose subsequent lines are indented. To form a hanging indent, type in part of the first line until the cursor is past the first tab setting. Pause to enter the ^OG command, then continue typing: the remainder of the paragraph will be indented.

Reformat a paragraph as a hanging indent in the same way: move the cursor past the tab setting to which the indent will be moved; enter the ^OG Indent command; then enter the ^B reformat command.

Hyphen Help

^OH *Hyphen Help ON/OFF Toggle*

WordStar normally asks for hyphenation decisions. This function can be turned OFF by entering the Hyphen Help command toggle ^OH.

When Hyphen Help is OFF, WordStar will not attempt to hyphenate words, but will reformat the paragraph by bringing words down to the next line when necessary. Hyphen Help is a toggle switch: turn it back ON by entering the ^OH command a second time.

Without Hyphen Help, some lines may have a lot of space in them to spread the words across the page for right justification. Some prefer Hyphen Help OFF during early draft stages, turning it back ON for the final draft.

Ragged Right (Justify)

^OJ *Right Justify ON/OFF Toggle*

WordStar normally right-justifies all text — entering soft spaces where necessary to make the right side of paragraphs line up evenly.

Right Justify can be turned OFF by entering the ^OJ toggle. When it is OFF, the right side of paragraphs will end near the right margin as with an ordinary typewriter. Non-justified text is sometimes called RAGGED RIGHT. This WordStar manual is composed with a ragged right margin.

Right Justify is a toggle command: turn it back ON by entering ^OJ a second time. Hyphen Help operates in the usual way when Right Justify is OFF.

When Justify is OFF and ragged right margin is in effect, no soft spaces are inserted in the text. Further, reformatting will remove all soft spaces from a paragraph if Justify is OFF.

Left Margin

^O L *Set Left Margin*

To set a new left margin, enter this command. WordStar asks you for a new left margin value.

You can enter a **number** for the left margin, then press RETURN. Default value for the left margin (its normal setting) is 1. The WordStar line is 128 characters wide.

Or you can press the ESC key; WordStar will set the left margin at the present cursor position. You must have the cursor already at the point where you want the left margin before starting the ^OL command.

There is no need to change the default left margin to ensure correct printout. The left margin value of 1 includes an eight-space left "gutter" or automatic left margin that indents the printer about one inch during printing. If you change the left margin setting, it is to indent the text still farther.

Right Margin

^[O][R] *Set Right Margin*

To change the right margin setting, enter ^[O][R]. WordStar asks for a new right margin value. The default (normal) right margin setting is 65.

You may enter the number you wish for the right margin and press RETURN.

Or you can press ESC to enter the current cursor position as the right margin. The cursor must already be at the point desired before initiating the ^OR Set Right Margin command.

Maximum right margin setting is 240 characters. Screen width is 80 characters. If the right margin is set past 80, when your typing reaches the right side of the screen, the screen's view will shift to the right 20 characters so you can see what you type. Every 20 characters thereafter, until the right margin is reached, the screen's view of the page shifts to the right to keep the cursor in sight.

If your margin settings are too wide for your printer, excess text may spill over from each line onto a new line, creating irregular printouts.

Quick Tip:

AUTOMATIC REPEAT REFORMATTING

The Reformat command ^B reforms one paragraph at a time. An entire document can be reformatted automatically by using the Reformat command together with the Quick Repeat command ^QQ.

To use, simply precede the ^B command with the ^QQ command: ^Q Q ^B.

The ^QQ command, which means "repeat the next command indefinitely," causes the ^B command to be reissued after each paragraph.

The command can be cancelled by pressing the spacebar. Otherwise reformatting continues until the end of the document. When the end of the document is reached, press the spacebar again to turn off ^QQ.

Once reformatting has begun, press a number between 0 and 9 to speed up or slow down the speed at which reformatting takes place: 0 is the fastest speed and 9 the slowest.

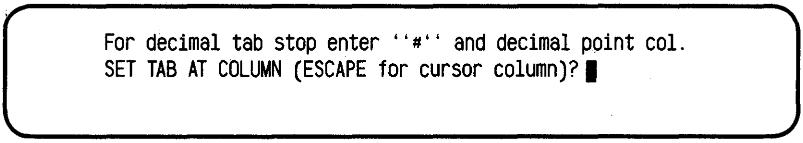
Some side effects of automatic reformatting: if a Hyphen Help decision is called for, the Quick Repeat command will be cancelled. It is therefore advisable to turn Hyphen Help OFF. Reformatting will reform every paragraph to current margins and current tab settings, so if you had indented paragraphs, they will be unindented. Reformatting has unpredictable effects on centered lines.

Tabs: Set & Clear

^O I or ^O TAB *Set Tab*

WordStar automatically sets a tab every five spaces. The ^OI command lets you set tabs in other locations.

Enter ^O I and WordStar asks:



```
For decimal tab stop enter ''#'' and decimal point col.
SET TAB AT COLUMN (ESCAPE for cursor column)? ▮
```

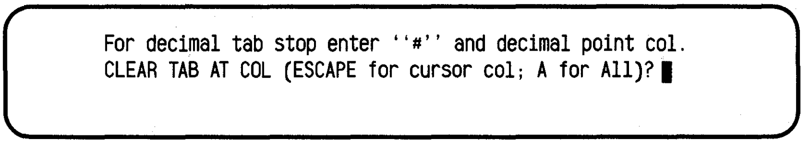
Enter a tab setting by typing the number of the tab location desired, then press RETURN. The tab is set, as indicated by a **!** on the ruler line at the location requested.

Alternately, you can place the cursor at the location desired, enter the ^OI Set Tab command, and then press the ESC key instead of entering a number. The tab is set at the cursor location, as indicated by the **!** on the ruler line.

^O N *Clear Tabs*

This command allows removal of specific tab settings or all tab settings.

To clear a tab setting, enter ^O N. WordStar asks:



```
For decimal tab stop enter ''#'' and decimal point col.
CLEAR TAB AT COL (ESCAPE for cursor col; A for All)? ▮
```

Enter the number of the existing tab setting to be cleared.

To clear ALL tab settings, enter **A**.

A tab setting can also be cleared by moving the cursor to the tab location, entering the ^ON Clear Tab command, and pressing the ESC key. The tab setting at the cursor location (if any) will be cleared.

The default of a tab setting every five spaces can only be reset automatically by exiting and restarting WordStar.

Decimal Tabs

^O I # *Decimal Tab Set*

Decimal tabs are used for entering columns of numbers so they line up properly.

When you tab to the location of a decimal tab setting and start typing, the cursor remains at the tab position, while the number you type is pushed to the left.

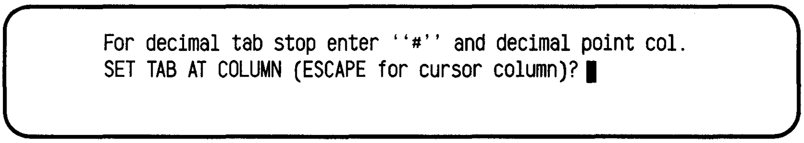
This continues until you type a decimal point (. - a period) or a space. Thereafter, whatever is typed is entered to the right of the decimal point or space, as in normal typing.

The result is that numbers in a column are properly lined up along the decimal point. Here is an example using two identical columns of numbers, one typed using Decimal Tab, the other using a regular tab setting:

```
With Decimal Tab           With Regular Tab
        *                        *
       34.75                     34.75
    $1,474.13                    $1,474.13
        1                        1
       12                        12
      123                        123
```

The decimal tab lines numbers up as you type. With a regular tab, you would have to add the right number of spaces to make everything line up.

To SET a decimal tab, enter ^O I, Insert Tab. Follow the instructions given in the on-screen WordStar prompt:



```
For decimal tab stop enter ''#'' and decimal point col.
SET TAB AT COLUMN (ESCAPE for cursor column)? ▌
```

by typing the number sign (#) before entering the column number for the tab setting.

As with regular tabs, you can also use ESC to set the tab at the cursor's location, after entering the #.

The decimal tab location is shown on the ruler line by a # instead of by the "!" used for regular tabs.

When you tab to the decimal tab location, the word "decimal" appears on the top line of the screen. Anything typed at that location will follow the decimal tab rules described above. This **includes** text, so remember to delete the decimal tab when it is no longer needed.

It is not necessary for the numbers being used to have decimal points. For round (nondecimal) numbers, set the decimal tab one character to the right of the last digit of one of the numbers. Notice the whole numbers in the example above and their location relative to the decimal tab setting.

Clear a decimal tab with the ^ON Clear Tab command in the same way as a regular tab (no # is needed).

On-Screen Layout

Advanced Level

Margins
Wordwrap
Fixed Tabs
Special Screen Displays

More Margins

^O F *Use Existing Margins*

This command changes the left and right margins to match those of an existing paragraph. This is useful for editing documents with frequent margin changes.

Move the cursor to a line with text running from margin to margin. Then enter ^OF. The left and right margins will reflect the width of that line.

^O X *Margin Release Toggle*

This command disengages the left and right margins. Repeat the command to end Margin Release.

Use this command to type past the right margin setting. It can also be used as an alternative to Paragraph Indent for creating a hanging indent: set the left margin at the indent column, then use Margin Release to type headings to the left of that setting (this was the method used for this manual).

Fixed Tabs

^O V *Variable Tab - Fixed Tab Toggle*

Variable tabbing is the norm when a file is opened with the D=Document command. Variable Tabs are automatically set every five spaces and can be removed and re-set elsewhere using the Tab Clear and Tab Set commands.

When the TAB key is pressed while Variable Tab mode is in effect, WordStar inserts spacebar characters in the gap. There is no "TAB" character as such: one edits a Tab in text by editing the five spacebar characters.

Fixed Tabs are the norm when a file is opened with the N=Non-Document command. Fixed Tabs are automatically set every 8 spaces and cannot be deleted or re-set at different positions: they are fixed.

Fixed Tabs are actual characters: the 8 spaces are not filled with spacebar characters, and the cursor cannot be moved into the tab area in text. Fixed Tabs are edited like RETURNs: delete the Fixed Tab character and the entire tab area is erased.

Fixed Tabs are used by programmers who wish to save program and memory space.

Use the ^OV command to toggle between Variable Tabs and Fixed Tabs. Toggling to Fixed Tabs does not change the appearance of the Ruler Line at the top of the page: it still reflects the settings of the Variable Tab mode, but the Tab key reacts to the Fixed Tab settings. Toggling back to Variable Tabs returns you to those settings, including any changes made in the tab settings.

Wordwrap Toggle

^O W *Wordwrap ON/OFF*

Wordwrap is normally ON in Document Mode and permanently OFF in Non-Document mode. In Document Mode, the ^OW command turns Wordwrap OFF, and when repeated turns it ON again. (It cannot be turned ON in Non-Document mode.)

Wordwrap automatically brings down to the next line words which extend past the right margin setting. When Wordwrap is turned off, you are obliged to provide the RETURN at the end of your line yourself. There is no bell or right margin lock to indicate when you have passed the right margin.

Wordwrap is sometimes turned OFF for the purpose of entering charts, special diagram material or headings, so as to prevent both the RETURN and the introduction of soft spaces into the line.

A line written with Wordwrap OFF can be reformatted using the ^B reformat command, whether Wordwrap is ON or OFF. RETURNs you entered to end lines while Wordwrap was OFF are hard RETURNs for reformatting purposes.

Wordwrap is OFF in the Non-Document mode so the non-ASCII representations of soft RETURNs and soft spaces will not be introduced into programming material.

Screen Display Toggles

^OT *Ruler Line Display ON/OFF*

The ruler line display can be turned off and back on again with the ^OT command.

^OD *Command Display ON/OFF*

This command turns OFF the on-screen display of certain WordStar commands that are normally embedded in text. The Print codes created by the ^P group, which are normally displayed in text (e.g., Boldface Print (^PB) displayed as ^B; Underline (^PS) displayed as ^S); and all soft hyphens created by Hyphen Help, are not displayed.

Since embedded print codes and mid-line soft hyphens make line endings appear ragged, the ^OD command is used to view text as it will actually print out. It is also useful for checking that columns of numbers or lists line up properly on the screen.

Only the display of these codes is banished from the screen: all commands are still active.

^OP *Page Break Display ON/OFF*

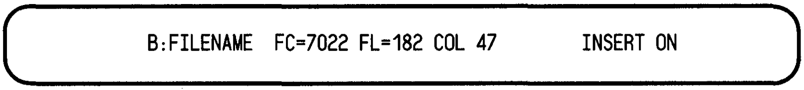
In a Document file, the end of a page is indicated by a string of dashes across the screen, with the letter "P" in the right flag column.

The ^OP command turns off the display of this page break line. Repeating the command turns the displays back on.

Page break display is permanently OFF when editing in Non-Document mode.

In a Document file, when page break display is ON, the top line of the screen displays the page number, line number, and column number of the current cursor position.

When in a Non-Document file, or when page break display is turned OFF while editing a Document File, the information displayed on the top line is similar to the following:



```
B:FILENAME  FC=7022 FL=182 COL 47        INSERT ON
```

FC=xxxx - Displayed here will be the number of "FILE CHARACTERS" in the document. File Characters consist of every character from the beginning of the file to the present cursor position, including spaces and RETURNs. Move the cursor to the end of the file to determine how much memory is consumed, at the rate of one character per byte of memory. For text files, divide the number by six for an approximate file word count.

FL=xxx - number of LINES in the document up to the cursor position will be displayed here.

Note:

Status of the toggle commands can be seen on screen by entering ^O. The On-Screen Menu is displayed, showing the status of wordwrap, ruler line, justify, variable tab, hyphen help, soft hyphen entry, print control display (command display), and page break display.

Special Lettering

Basic Level

Underlining   *Boldface*

Underlining

^P S *Underline ON/OFF*

To underline a word, do not use the underline stroke key (—). Instead, enter the Underline command **^PS** once before the first word, and again after the end of the word or phrase to be underlined.

The symbol **^S** is displayed on screen to show where the underline begins and ends. It makes the line appear to go past the right margin; it prints out justified, however.

You must bracket the item to be underlined because the command is a **toggle**; using it once turns underlining ON; it stays on, underlining every printing character, until used a second time to turn underlining OFF.

This command underlines characters but not the spaces between characters. To get a continuous underline, use the Underline key on the keyboard between words (which will interfer with Wordwrap, so use it with caution on multiple lines).

Boldface

^P B *Boldface ON/OFF*

This command causes the printer to overstrike the item three times, making it appear darker.

To emphasize the effect, on printers capable of incremental spacing each of the three strikes is offset slightly, making fatter as well as darker characters.

This is a toggle switch and must bracket the character, word or phrase you wish boldfaced. ^PB stays ON until a second ^PB turns the effect OFF.

The screen displays **^B** to show the presence of the command. The paragraph will appear to have an uneven right margin, but will print out justified.

Special Lettering

Intermediate Level

Double Strike
Hard Hyphen
Non-Break Space
Superscript and Subscript

Double Strike

^P D *Double Strike ON/OFF*

Double strike is similar to Boldface, except that the printer strikes twice instead of three times, and does not use offset printing.

Double strike, which is also a toggle command and must be repeated to be turned OFF, is sometimes used with a carbon ribbon to obtain a crisper printed copy.

^D is displayed on screen to show this toggle command.

Non-Break Space

^[P][O] *Non-Break Space Character*

This command enters a "hard" space between words such that WordStar will not insert additional soft spaces or a soft hyphen.

For example, in the phrase "May^O17th, 1983," the use of the command ^PO instead a spacebar will ensure that the word "17th" will not be brought down to the next line by Wordwrap.

The command ^PO, when it is the only form of space between two words, will also prevent WordStar from adding soft spaces between two items. For example, in numbered paragraphs the number of spaces between the number and the first word of each paragraph will vary as soft spaces are added to justify the line. This can be prevented, and the paragraphs aligned properly, by using Non-Break spaces instead of ordinary spaces between the number and the first word of each paragraph.

The Non-Break Space will only work if it is the only kind of space dividing the two items.

Non-Break Space is displayed on screen as ^O.

If there are insufficient other words in the line to allow proper right-justification, WordStar will leave the line unjustified.

Hard Hyphens

^O E ***Soft Hyphen ON/OFF***
^P - ***"Hard" Hyphen Character***

These two commands are used together. Hyphens (dashes) you type are normally "hard" hyphens — they will always print out. When reformatting (^B), hyphens entered in answer to Hyphen Help are "soft" hyphens, are displayed on screen at half intensity, and are only printed if they occur at the end of the line. If reformatting puts the word back together on one line, the soft hyphen will still display on screen at half-intensity, but will not print.

When the ^OE command is entered, all hyphens typed in subsequently are soft hyphens, even ones you type in while entering text. This command might be used to predivide long words or foreign terms which the Hyphen Help program might misdivide by applying English-language hyphenation rules.

While the ^OE command is in effect (it is a toggle and must be switched off when done), you must use the ^P- command if you need a hard hyphen.

The status of the ^OE command (default is OFF) can be determined by viewing the ^O Menu.

Superscript and Subscript

^P V *Subscript ON/OFF*

If your printer is capable of printing subscripts, this command will toggle subscripting ON and OFF.

An example of a subscript is the ''2'' in H_2O.

Enter ^PV where the subscript is to begin. A second ^PV ends the subscripting and returns the printer line to normal. (The Superscript command will also end subscripting if the increment values for both are the same.)

The presence of this toggle is indicated on screen by the symbol ^V.

^P T *Superscript ON/OFF*

This command initiates superscripting, if your printer is capable of superscripting. A second ^PT must be entered to end superscripting.

An example of superscript is the ''2'' in 10^2.

This toggle is indicated on screen by the symbol ^T.

Note:

On printers that do not have incremental scrolling, the superscripts and subscripts will roll up or down one full line. You must doublespace to allow for this, or the super/subscript will overprint another line. On other printers, the super/subscript command will be ignored.

Special Lettering

Advanced Level

Alternate Pitch
Stop Print
Strikeout
Special Print Wheel Characters
Miscellaneous Control Characters
Set Pitch
Overprint
Ribbon Color
User-Defined Functions

Alternate Pitches

^P A *Use Alternate Character Pitch*

^P N *Restore To Standard Character Pitch*

Pitch is the number of characters printed per inch of text. Some printers offer two or more pitches. Most daisy wheel printers, for example, offer 10 pitch (pica) as standard pitch and 12-pitch (elite) as an alternate pitch.

Normally, 10 pitch is used. But the alternate pitch may be selected by entering the command ^PA into the document at the point where the 12-pitch printing is to begin.

This alternate pitch printout continues until the Standard Pitch command ^PN is encountered in the text; printout is returned to standard pitch.

If the printer is turned off and then on, the 10-pitch default again takes effect.

Note:

It may be necessary to change the print wheel to take full advantage of alternate pitch. Use the Stop Print command described later to allow the change.

Pitch is controlled by sending a code, called the printer initialization string, to the printer. The strings built into ^PN and ^PA normally signify 10 pitch and 12 pitch to a daisy wheel printer. You can use the .CW dot command (see **Set Pitch** below) to assign other pitches to the ^PN and ^PA commands.

^PN and ^PA usually have no effect on dot matrix and other non-daisy wheel printers, even with use of the .CW dot command. However, if your non-daisy wheel printer offers more than one character pitch (or FONT), you can use the WordStar Install procedure (explained elsewhere in this guide) to assign the required printer initialization string to the ^PA and ^PN commands so they will work on your printer.

Set Pitch

.CW4 - 24 ***Set Character Pitch Value***

This command sets the pitch to be used by the ^PN and ^PA print commands described above.

Default character width value is 12 for "standard pitch" (^PN—equivalent to 10 pitch, see table below) and 10 for "alternate pitch" (^PA—equivalent to 12 pitch).

Character width is set in 1/120ths of an inch. Use the following table to determine the character width number for the command to get the desired character pitch.

Pitch (Char./inch)	.CW "n"
5	24
6	20
7	17
8	15
10	12
12	10
15	8
20	6
24	5
30	4

This command must be on a line by itself and must be the first characters on the line, beginning in column 1. No spaces may precede the command and no text or other commands may follow it, since WordStar ignores any commands or text on the same line as a "dot command" (WordStar commands which begin with a period).

To use the Character Width command, type it in the text at the point where you wish it to take effect.

The command affects the following line; it stays in effect until the end of the file or until the next character width command.

The .CW command changes the character width value of the Pitch command that is in effect at that point in the text. For example, if Standard Pitch is active (if ^PA Alternate Pitch has not been entered by that point in the document, Standard Pitch is active by default), then the .CW Set Character Width command affects the value of the Standard Pitch command. It does not affect the value used by the Alternate Pitch command.

Either Standard or Alternate Pitch or both can be altered within a document as needed. Enter the Pitch command which must be changed, followed by a Character Width command to change its value. Then enter the next Pitch command, followed by its Character Width command value.

The Character Width command only works with printers having programmable character widths (usually daisy wheel printers). Dot matrix printers cannot usually react to the Character Width command: use a printer initialization string (see the WordStar Install program instructions).

It may be necessary to change print wheels for best effect. See the Stop Print command ^PC for this purpose.

Quick Tip:

THE DIFFERENCE BETWEEN "SET PITCH" AND "ALTERNATE PITCH"

To clarify the difference between the "set character pitch" dot command ".CW" and the "print character pitch" print commands ^PA/^PN:

- *The dot command* ***.CW*** *is used to* ***define the value*** *(pitch) of either or both of the print commands.*
- The print commands ***^PN*** *and* ***^PA*** *instruct the printer to* ***begin using*** *the value in the document.*

If no Character Width command is issued, the print commands use their ***default values*** *of 10 pitch and 12 pitch (characters per inch), corresponding to a character size (width) of 10/120ths and 12/120ths, respectively.*

If no Print Command is issued, the document is printed out in the default of 10 pitch.

The .CW dot command, which must be on a line by itself, takes effect on the following line of the document. The print commands ^PA and ^PN can be in the middle of a text line and take effect on the next character of that line.

It is possible to use the Character Width command to set the values of the print control commands at the very beginning of a document, then issue the ^PA and ^PN commands as needed in the document. It is also possible to change either or both values in mid-document.

Special Print Commands

^PC *Stop Print*

The ^PC command causes the printer to stop printing at a given point in the text.

This command is commonly used to stop printing so the print wheel or the ribbon can be changed.

"Print Paused" appears on the computer screen until the printer is restarted by entering P (from the Opening Menu) or ^KP (if printing while editing a file).

Insert the command at any point in the document where the printer should stop. It can be used as often as desired, even several times on the same line. For example, to print something in italics, bracket the item in ^PC commands. During printout, the printer will stop before the word; change the print wheel and restart with P (or ^KP). The printer will stop again after printing the italicized item so you can change the print wheel again.

^PH *Backspace and Overprint*

This command causes the printer to back up one character, then continue printing.

The result is that the printer will overprint one character on top of another. Accents and other diacritical marks on foreign words can be built up with this command.

The symbol **^H** appears on the screen to mark the location of this command.

^P X *Strikeout ON/OFF*

Using this command causes all the text between two ^PX commands to be printed with dashes through them.

The Strikeout command is commonly used to show deleted text when revising certain kinds of legal papers.

^P RETURN *or* ^P M *Overprint Next Line*

This command is used at the end of a line in place of an ordinary RETURN or ^M command. It causes the following line to print on top of the line that ended with this command.

The RETURN key normally inserts in the text a RETURN to the left margin plus a Line Feed to the next line. This command, however, inserts only a RETURN without the Line Feed.

On the screen the lines will be one after the other, but a dash appears at the far right side of the screen in the Flag Column.

^P Y *Ribbon Color ON/OFF*

For printers with selectable ribbon color, this toggle will cause the text following the command to be printed using the alternate color on the ribbon.

Using the command a second time toggles the printer back to the standard ribbon color again.

Special installation may be required for non-daisy wheel printers for this effect to work properly.

Special Print Wheel Characters

^P F ***Phantom Space***
^P G ***Phantom Rubout***

Some daisy print wheels have additional or alternate characters. An alternate character may be accessed by typing the character to which it is an alternate. Added characters may be accessed using the Special Print Wheel Character commands.

^PF, called "phantom space," prints the daisy wheel character corresponding to hex code 20. This is usually interpreted as an ordinary space on non-daisy printers (hence the name), but prints an alternate character (most commonly a "cents" sign) on some 88- and 96-character print wheels. Exactly what is printed depends on the print wheel you use and can be determined by experimentation.

^PG, called "phantom rubout," prints the "not-sign" or "double underline" or other character corresponding to hex code 7F on many 96-character daisy print wheels.

User-Defined Functions

^P Q ^P W ***Extra Function Keys***
^P E ^P R ***Definable By User***

WordStar provides four extra unassigned function keys to which you may assign functions of your own devising, using the Install procedure outlined elsewhere in this guide.

The Install procedure involves entering a special "hex code" on the WordStar program disk to access the function you wish to use. Special function keys activate extra capabilities of your printer: continuous underlining (instead of character underlining that skips the spaces), for example, or italicized characters on dot matrix printers.

The printer manual will list the hex codes that you must enter to obtain to these special functions.

Your newly assigned function keys are used in the same way as regular function keys. Some will be "toggles," which are used once to turn the special function on, and a second time to turn the special function off. Some functions will be a matched pair, like the Alternate Pitch function, which uses one command to access the function and another separate command to turn it off. The printer manual will usually designate toggle functions.

Miscellaneous Control Characters

^P L *Form Feed Command*

This command causes a page break in the same way as the dot command ".PA", except that the letter L followed by a row of dashes is produced (instead of a row of dashs followed by the letter P).

^P I *Fixed Tab Key*

This command inserts a tab corresponding to the location of the Fixed Tabs. See the discussion of **Fixed Tabs** elsewhere in this section.

This command is not normally used since the TAB key will perform the same function while in Fixed Tab mode. However, this command will insert a tab to the Fixed Tab setting even while in Variable Tab mode.

^P J *Line Feed*

This command is identical to the RETURN key or the ^M command, entering a line feed and carriage return.

Although on screen the line feed will go down one line and over to the left margin, a "J" in the flag column shows this as a line feed, not a true RETURN. On the printer, the printing head will drop down one line and continue printing, but will not go to the left margin.

Print Controls

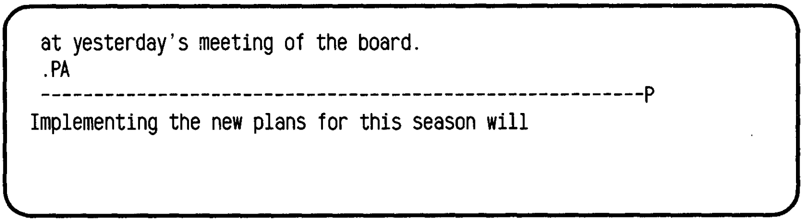
Basic Level

Page Break
Headings
Page Numbering
Footings

Print control commands affect how the printer handles the printout of your document. Special codes on the screen show when print controls have been used, but their full effects are only visible on the printed page.

Print control commands specify such factors as page numbers, headings and footings, and the exact size of page margins.

Dot commands are the principle type of printout controls. These commands are preceded by a period or dot rather than the CTRL key. Here is an example of a dot command:



```
at yesterday's meeting of the board.
.PA
---------------------------------------------------------P
Implementing the new plans for this season will
```

This dot command, .PA, highlights some key features in the use of dot commands:

1. The command must start at the left margin, in column 1 (even if the left margin is indented). The period must be the *first* character in that line — not even a space must come between the first column and the dot command. Watch out when reformatting: if a dot command becomes indented, it won't work.

2. The dot command is not printed. Nothing on the same line as a dot command is printed (except the text of a heading or footing, described below). If you find that a text line has failed to print, check to see if a period got into column 1 of the line.
3. The dot command can be upper or lower case.
4. WordStar doesn't count dot command lines when figuring page lengths.
5. You can use several dot commands at the same time; but each must be on a separate line.
6. A dot command remains on for the rest of the document or until it is turned off by another dot command (or the same dot command if it is a toggle).
7. Dot commands can be deleted or block moved just like any other characters.

Some dot commands (line height, page length) affect line count on the page. If they are put in the middle of the document instead of at the beginning, WordStar will display the following next to the dot command:

```
          ''Put at file beginning for correct page break display.''
```

It means that page endings displayed on screen may be different from the actual page endings in printout. The dot commands will still work properly during printout: it is the screen display that is affected.

Start New Page

. P A *Page Break*

This dot command causes WordStar to start a new page.

As with the ordinary page breaks that WordStar inserts every 55 lines, a dotted line is inserted across the screen, with a **P** in the flag column, to show where one page ends and the next begins.

When you move the cursor down below the page break marker, the page number and line number readout in the menu changes to reflect the new page.

Always put a RETURN after the page break command. This is especially important when putting a page break command at the end of a file: if you neglect the final RETURN, succeeding files printed out with the M=Merge-Print command will lose their first line of text and not print out properly.

The page break command is useful for separating a series of files you are printing using the Merge-print command, which is described in a later Advanced section as a method of printing a series of files. Without the page break, WordStar runs the files together, starting the next file on the same page as the end of the previous file.

Numbering Pages

. O P *Omit Page Number*

WordStar always prints a page number, at the bottom center of the page. For a one-page document you may not want a page number printed. Put this command at the top of your document to accomplish this.

.PN *Start Page Number*

This command will turn the WordStar page numbering function back on.

The pages following this command will be numbered correctly taking into account the number of pages preceding them in the file.

This and the .OP command can be used together: .OP to prevent the first page of a document from being numbered; .PN to turn the numbering on again beginning with the second page. To be effective, the .PN command must be put on the **preceding** page (anywhere on the page after the first line of text).

.PNn *Start Page Numbering With n*

You can start the numbering at any number you wish, by substituting the number for the "n" above, thus: .PN 5, which starts page numbering with the number 5.

When printing a file that is a continuation of another file that was printed separately, this command will let you start the file with the correct page number (otherwise WordStar would start with "1").

Headings and Footings

.HE *text* # *Page Heading*

(The # is optional; see below.)

A heading is a text line that is printed across the top of every page of a document. A footing is a text line printed across the bottom of the page. WordStar will automatically repeat your heading and footing on each page.

To start a heading, enter the dot command .HE at the left margin. Then type in the text of the heading. An example:

```
.HE                              OFFICIAL HANDBOOK
```

The heading is printed on each succeeding page, two lines above the page text. No space is taken from your regular text area. The heading will appear in printout only, not on the subsequent pages on screen.

In the example, the phrase "OFFICIAL HANDBOOK" will appear on the upper right side of the page (because of the spaces that have been inserted between the .HE command and the phrase). To have it print out on the left side of the page, eliminate the spaces.

To center the phrase, type in the phrase and use the ^OC centering command; once the phrase is centered, go back and insert the .HE command beginning at column 1. Do not try to center the entire heading command since this will move the dot command away from column 1, rendering it inactive.

To include the page number in the heading, use the symbol #, thus:

```
.HEOFFICIAL HANDBOOK                PAGE NO. #
```

The page number will appear on each page at the # mark. It can be placed anywhere in the heading line.

To have the # mark itself print instead of a number, precede it with a ^P\, thus: ^P\#. This "print as character, not as code" command is good only for the one character that follows it. If you type ^P\##, the first # will print as a #, but the second # will be replaced by a number.

The heading will be repeated on each page until replaced by another heading, or cancelled by the heading command with no text following it.

Headings are one line long. There are no multiple-line headings. If you enter more than one line as a heading, only the most recent heading line will be printed.

A page number is always printed at the bottom center of the page, whether or not the heading command includes a page number symbol. To prevent this duplication of the page number, enter the .OP, Omit Page, command as shown in the example below.

```
.PN 12
.HE  OFFICIAL HANDBOOK                  PAGE NO. #
.OP
```

In this example, the page number, which will begin with the number 12, will appear in the heading but not on the bottom of the page.

Each dot command takes a separate line.

For the heading to print out on the **first** page of the document, the heading command must precede everything in the document file (except other dot commands). If it follows any text, including RETURNs, the heading will begin on the next page.

.FO ***text*** # ***Page Footing***

(The # character is optional; see below.)

A footing is the same as a heading except it appears at the foot of the page, two lines below the bottom of the main body of the text. Like the heading, it takes no lines away from the main text space. It is printed on the bottom of every page following its entry, but it is not displayed on every page on screen.

The rules for footings are the same as for headings. If a footing is used, page numbering is turned off (unless you add the # character to the footing line) since the footing appears on the line the page number would normally use. Therefore it is not necessary to use the .OP command as with headings.

Both a footing and a heading can be used at the same time; both can have page numbering.

Insert the footing command at the top of the first page of the file, in the same way as a heading command, not at the bottom of the page; WordStar will take care of positioning it correctly at the bottom of each page. If the footing is to start on other than the first page of the file, it is entered on the preceding page, in the same way as a heading command.

Print Controls

Intermediate Level

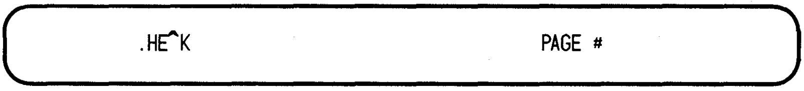
Alternating Headings	*Alternating Footings*
Conditional Page Break	*Page Number Location*
Comment Line	*Microjustification*

This section explains some further refinements in how the page will be printed.

Alternating Headings & Footings

^P K *Alternating Headings & Footings*

This command is used **within** the heading or footing command line to cause the heading or footing text to print out on the right side of odd-numbered pages and on the left side of even-numbered pages. Here is an example of its use:



```
.HE^K                                    PAGE #
```

Because of the ^PK command, on even-numbered pages WordStar will ignore all the spaces between the ^PK (displayed on screen as ^K) and the word "PAGE".

As a result, "PAGE #" will show up against the left margin of the even-numbered (left-hand) pages. On the odd-numbered (right-hand) pages, the spaces are left in so that "PAGE #" appears on the right side of the page.

This command is used in headings or footings or both. The command is entered within the actual heading or footing, directly before the spaces that will be deleted on the even-numbered pages.

Page Number Location

. P C n *Page Number At Column "n"*

The page number usually appears centered at the bottom of the page (unless the # is used in a heading or footing). Use .PC plus a number to place the page number on the bottom of the page other than in the center.

The number represented by the "n" is the number of columns (characters) to the right of the left side of the page the page number is to appear. A number corresponding to the left margin setting ("1" unless you have indented the left margin) would put the page number flush left. A "65" would place the page number on the right margin if your right margin is set at 65.

Conditional Page Break

. C P n *Conditional Page Break "n" Lines*

This command is used to keep a block of text together on one page. It prevents WordStar from putting part of a chart on one page and the rest on the next page; or putting a section title at the bottom of the page and the text on the next.

The .CP "n" tells WordStar not to put a page break anywhere in the next "n" lines. If a page break situation arises, WordStar is to start a new page instead, so that the block of "n" lines remains together.

For example, to ensure that a table that is 15 lines long is kept on one page and not split between two pages, put the command **.CP 15** on the line above the top of the table (like all dot commands, in must be on a line by itself). This turns the next 15 lines into an unsplittable block of text.

If at any point during reformatting or printing there is not enough room, on the page being reformatted or printed, to fit the entire 15 line block, WordStar starts a new page, even if that means leaving a 14-line gap at the bottom of the page.

The new page is shown on screen with the page break line just like a normal page break. If the file is reformatted, WordStar ensures that page breaks follow the orders of the .CP commands.

This command can be used throughout a file to keep charts, tables, illustration spaces, and titles in one piece. Putting .CP 4 above a subhead, for example, will guarantee that at least two lines of text will follow on the same page, or else the subhead will be moved to the next page. A .CP 2 above every paragraph ensures that the first line of any paragraph won't be orphaned on the bottom of a page.

Micro-Justification

[.] [U] [J] [0] ***Microjustification OFF***
[.] [U] [J] [1] ***Microjustification ON***

You may find a discrepency between the way words line up on screen and the way they are printed out. A chart which was visually lined up on screen may print out misaligned. The reason is microjustification.

Justification adds soft spaces between words so that the right side of the text is right justified, or evenly lined up.

The screen, and many printers, add whole spaces, the size of normal characters, between words to achieve the even right margin. When material you visually aligned on screen is run off on a printer that uses whole characters for justification, the material will line up exactly as seen on screen. A close inspection of the printout will reveal that some words are separated by one space and some by two.

More advanced printers are capable of incremental spacing. They convert WordStar's whole-character soft spaces into incremental spaces 1/120th of an inch in size, and distribute these spaces evenly among all the words on the line. This "microjustification" improves the appearance of the printout. However, it also throws the material you lined up out of alignment during printout.

This can be prevented with the .UJ commands. On the line above your table, chart, form, or other aligned material, enter the command .UJ 0. This will turn the printer's microjustification function OFF.

On the line following the columnar material, enter .UJ 1 to reset microjustification for the rest of the document.

The text between the commands will be printed exactly as it appears on the screen.

Note that this command doesn't turn off WordStar's Wordwrap and Right Justify: it just prevents the printer from converting WordStar's soft spaces into microjustified spaces.

Entering Tables and Charts

Tables and charts demand different qualities of the word processor than text. Wordwrap can be a hindrance if a columnar entry is brought down to the next line by accident.

Here are some settings to use when entering columnar material:

1. Enter ^OW to turn Wordwrap OFF to avoid soft spaces.
2. Enter .UJ0 at the beginning of the table to turn microjustification OFF.
3. Enter ^OJ to turn Right Justification OFF; or else move the right margin well off to the right with the ^OR command.
4. Clear all tabs, and reset regular and decimal tabs where desired. See the Quick Tip below for another method of setting tabs.
5. Alternately, write each column of numbers/words out separately and Column Block Move each column into position on the chart. With the Column Block Move command, the column position can be easily adjusted.

To edit an existing chart:

1. Set tabs as shown in the Quick Tip below.
2. Turn Insert OFF with ^V.
3. Now you can TAB over to each column without inserting spaces. A number can be typed over and replaced.

4. When writing multiple headings or titles to columns, put in print controls like boldface and underline AFTER the headings are completed. Otherwise the print symbols will make visual on-screen alignment difficult. Alternately, turn print control display OFF momentarily with ^OD to confirm heading alignment. Then turn it back on again.
5. The underlining command, ^PS, underlines characters, not spaces. To create a solid underline for a heading, type underline characters between words instead of spaces, like this: ^SYear_To_Date^S.

Quick Tab Resetting

Frequently-used charts with their own tab settings require frequent manual tab resetting. This is how to reset tabs quickly on existing charts.

1. When you have determined the correct tab settings, type a line similar to this on a line above the chart:

   ```
   L------------!-------------#-------#--------#---------------R
   ```

   The L and R correspond to the Left and Right margin being used in the chart; the exclamation point is a regular tab setting and the # character the location of a decimal tab. Dashes separate these characters to indicate no tabs. In the sample above, the tab symbols indicate exactly where tabs are located in the chart that follows.

2. Return to the left margin of the line the above sample is typed on. Type in two **periods** (dots), then CTRL-RETURN, thus: . . ^RETURN. The result will be an "overprint line" with two "comment dots" that prevent the sample tab line from appearing in your printout.

3. Finally, move the cursor to the sample tab line and enter ^O F - this is the command that causes margins to be set according to the size of the line the cursor is on.

   As it happens, this command also enters a regular tab where it finds a ! on that same line; a decimal tab when it finds a #; and erases any existing tabs where it finds a dash (-).
   This then is your command line for ordering up margins and tabs for a chart. When returning to this chart in the future, move the cursor to this command line and issue the ^O F command. Margins and tabs will be adjusted appropriately. A similar line below the chart can be used to return the tabs to their correct setting for the text.

4. For standardized charts, it is possible to keep the tab command line in a separate file on the disk, using ^KR to read in a copy into any new documents as needed to define tab settings, then deleted.

To enter a non-printing comment, first enter the dot command .IG ("IGnore") in column 1, then type in the desired remark. It will display on screen but will not print.

Alternately, use two periods starting in column 1, and WordStar will ignore the following text during printout.

WordStar in fact ignores any line that begins in column 1 with a period and doesn't contain a recognizable dot command, so any line preceded by one or more periods will fail to print, as long as the first two characters following the period don't make up a valid dot command.

Print Controls

Advanced Level

Page Offset
Heading & Footing Margins
Line Height
Bidirectional Print
Top & Bottom Margins
Paper Length
Superscript/Subscript Roll

Page Offset

. P O n ***Set Page Offset To "n" Characters***

This command "offsets" or indents the text from the left edge of the paper (or leftmost print position of your printer).

Normally, with the left margin set at "1" WordStar automatically offsets or indents 8 columns from the paper edge. Each column is the size of one character of whatever pitch is currently active. For 10 pitch, the result is a left margin just under one inch.

With the left margin set at 1 and the right margin at 65, this offset centers your text on 8.5-inch-wide paper.

To change this indent value, use the .PO, Page Offset, command. Enter the command at the beginning of the document file, insert the number of columns to be indented, and RETURN.

With this command you can create narrower or wider margins to center text that is other than 65 characters wide, or printed in other than 10 pitch, or printed on paper that is other than 8.5 inches wide.

This command together with the ^OL, Left Margin, setting determines the size of your left margin, while the ^OR, Right Margin, command determines the size of your right margin.

Top & Bottom Margins

. M T n *Set Top Margin To "n" Lines*

WordStar normally carriage returns three lines from the top of the page before printing.

Change this with the .MT command, specifying the number of lines ("n") to come down from the top before beginning the text.

How large a top margin is depends on how far down from the top of the actual paper your print head is set before printing begins. It is also affected by the .LH command discussed later in this section.

. M B n *Set Bottom Margin To "n" Lines*

WordStar normally stops printing eight lines from the bottom of the page. You can specify a different number of lines with this command by substituting the desired number of lines for the "n."

Heading & Footing Margins

. H M n *Set Heading Margin To "n" Lines*

When headings are used, they are printed within the top margin area, normally two lines above the body of the text.

When the default Top Margin setting of three lines is used, this means the heading is printed one line down from the top of the page and two lines above the body of the text.

This command sets the number of carriage returns between the heading and the body of the text. Don't make this number larger than the actual space available as set in the top margin, since the heading is supposed to fit within the top margin.

.FMn *Set Footing Margin To "n" Lines*

Footings are printed within the bottom margin area, usually two lines below the last line of the text. Use this command to select another number ("n") of lines below the last text line for the footing to appear. Don't make this number larger than the actual space available as set in the bottom margin.

Paper Length

.PLn *Set Paper Length At "n" Lines*

Paper length is assumed to be 66 lines per page. This corresponds to 6 lines per inch on 11-inch paper. (The number of actual lines printed is governed by other commands — this is the actual total length of the page including top and bottom margins.)

This command allows you to specify another paper length, in lines per page. This command is used if you are using paper of a different length (e.g., 14-inch legal paper), or if you change line height as explained below. If you change line height, calculate the correct paper length setting by multiplying "lines per inch" from the chart in the section below, by the paper length in inches.

Line Height

. L H n *Set Line Height Value*

Line height is the distance from line to line, measured in number of lines per vertical inch.

Line height affects the number of lines that can be printed on a page.

Normally WordStar assumes six lines per inch. The .LH command allows you to set another value, measured in units of 1/48th of an inch. Refer to the table below for the value to enter as "n" in the line height command:

Lines/inch	.LH value
2.0	24
2.4	20
2.6	18
3.0	16
4.0	12
4.8	10
5.3	9
6.0	8*
6.8	7
8.0	6
9.6	5

(*Normal default value)

The line height command can only be used by printers capable of incremental line spacing.

Setting the line height value to 12 or 16 at the beginning of a document is an easy way of getting a space-and-a-half or double-spaced draft of your document without having to reformat. The line height command can later be deleted to get a final single-spaced printout.

Superscript/ Subscript Roll

. S R n *Super/Subscript Roll Value*

This command is used to change the amount WordStar rolls the printer platen up or down for a superscript or subscript. The normal value is 3/48ths inch. With this command you can change that value in 1/48ths inch increments. See "Superscript/Subscript" in the Special Lettering section.

Substitute for "n" the number of 1/48ths inches desired. For example, the WordStar default value, 3/48ths, would be entered as .SR 3.

Bidirectional Print OFF/ON

. B P 0 *Bidirectional Print OFF*
. B P 1 *Bidirectional Print ON*

On most printers, the printhead or printwheel prints in both directions, from left to right on one line and from right to left on the return line, for maximum output speed.

The command .BP 0 will force the printer to print out only left to right.

This command is used when the printer or paper feed is not providing accurate enough printout alignment.

When bidirectional printing has been turned OFF, the command .BP 1 will turn it back on again.

File Handling

Basic Level

Save
Delete a File
SHOW
"Disk Full"
Quit
Rename
Copy
DIR

Saving the File

^K D ***Done Edit - Save File***

When you're done editing the document, this command closes the file and saves it to disk.

You then return to the Opening Menu, from which you can edit a new file or end your WordStar session.

Any earlier version of the same file that is on the disk is saved as a BACKUP FILE, which can be used in the event your edited file is damaged.

WordStar creates the backup file by changing or adding the extension **.BAK** to the existing file name. Examples:

MARCH3	is changed to	MARCH3.BAK
RIPLEY.LTR	to	RIPLEY.BAK
MAINDOC.12	to	MAINDOC.BAK

The edited file is saved under the file name you gave or used. In the above example, the previous version of MARCH3 is renamed MARCH3.BAK; the newly edited file is saved as MARCH3.

(Should you get a Disk Full message while saving, see the section on that subject below.)

^K S *Save and Re-Edit*

The ^KS command is used to save a temporary file while editing.

When the ^KS command is entered, WordStar saves the file you are editing to disk, then returns you to the beginning of the file so you can continue editing.

This lets you save your work as you go. Should your open file be damaged by a power failure or other cause, you can turn to a recent copy of the file.

The ^KS command is more convenient than the alternative of using the ^KD command to return to the Opening Menu, then using the D command to reopen the file for further editing.

It is strongly recommended that you use this command regularly — every fifteen minutes, or every one or two pages of text — to save your work. Should power problems occur, you will not lose more than a page or two of your work. Power supply stability (brownouts, line voltage spikes and drops) can be a noticeable problem in some areas, and the presence of heavy machinery on the power line as well as weather are potential sources of data loss. The ^KS command is designed to minimize your potential losses.

(It is also vital that backup copy disks be made of all data files on a regular basis to protect against loss due to disk damage and wear. See the COPY program described in *Volume 4 — Operating Systems* and also in the *Beginner's Guide*.)

After the ^KS command has been entered, you will find yourself at the beginning of the file. Use the command **^QP** to return immediately to the location you were editing in the file before the save. **^KS** and **^QP**, used in combination, let you save your document quickly and return to your position to continue creating or editing your document.

The ^KS command is preferable over the ^QR command as a way of moving a long distance (10 pages or more) backwards in a file, for reasons of data safety. A long backwards jump in a document can sometimes cause file scrambling. The ^KS command guarantees a safe re-entry at the file beginning.

^K Q *Quit a Document*

With this command you abandon the document you were editing, without saving a backup copy or altering the original document.

You are returned to the Opening Menu.

This command may be used if the text has become garbled due to power failure, if you have changed your mind about changes you have made, or when you are just examining a file without intending to make changes.

This command can be used as an "emergency rescue" command if you make a serious mistake while editing, or delete a large block and change your mind. You can always use the ^KQ command to quit the document, leaving the original version intact.

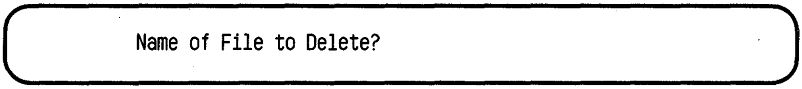
If you made any modifications of the original document, you will be asked "Abandon edited version of file? (Y/N)" to confirm your decision.

File Commands

These commands are listed in the Opening Menu under the title "File Commands." You must be in the Opening Menu to use them.

Y *Delete a File (From Opening Menu)*

When in the Opening Menu, press the letter Y, and WordStar asks:



```
Name of File to Delete?
```

Type the name of one of the files on the logged disk. Then press RETURN.

If the file to be erased is on the nonlogged disk drive, precede the file name with the disk drive and a colon, for example: A:MARCH.LTR. Don't forget the colon.

If you change your mind about deleting, use the Interrupt command (^U) **before** pressing RETURN.

Delete files one by one (you cannot use "wildcard" characters to delete several files at once, as in CP/M Plus). WordStar does not ask for confirmation of deletion, so be careful that you are deleting the correct file.

Some files are "read-only" files, and cannot be deleted by WordStar. These files are delete-protected by CP/M Plus using the SET command (described in *Volume 4 — Operating Systems*). If you try to delete a read-only file (shown on the DIR listing explained below as "R/O"), not only will WordStar refuse to delete it, but the program will usually end (or "crash"), returning you to the CP/M Plus operating system.

You will know you have been dropped out to the operating system because the symbol **A>** (if you are logged on the A drive) or **B>** (if you are logged on the B drive) will appear on the screen immediately to the left of the cursor (whatever was displayed on the screen at the time of the crash may still be displayed so you might have to look to find the A>).

To recover from this kind of crash, simply type in one of the following:

If the prompt is A>, enter W S RETURN;

if the prompt is B>, enter A : W S RETURN.

WordStar will reload and return you to the Opening Menu.

If you wish to erase all the files from an entire disk, consider using the CP/M Format program (see the CP/M Plus section of the *Beginner's Guide*).

Once a file is deleted, WordStar cannot recover it. (There are programs such as Disk Doctor which are able to recover erased files in some instances.)

E ***Rename a File***

From the Opening Menu, you can give an existing file a new name. Press the letter E. WordStar asks:

```
Name of file to rename?
```

Type the name of the existing file, then press RETURN.

WordStar then asks:

```
New name?
```

Type in another name, then press [RETURN].

The new name will replace the old one in the directory.

You can rename a file on the nonlogged drive by preceding your answer to the first question with the correct disk drive letter and a colon (A: or B:).

This command is the one you use to access a backup file. As explained in an earlier section, use E=Rename to give the .BAK file a name without the .BAK extension in it. The backup file can now be edited. (If you try to edit a backup file, WordStar reminds you of these facts by saying "Cannot edit .BAK file; rEname first.")

[O] *Copy a File*

When you are in the Opening Menu, this command lets you make an exact copy of a file, on the same disk under another name, or on another disk.

Press [O]. When WordStar asks:

```
Name of file to copy from?
```

Respond by typing the name of the file and press [RETURN].

WordStar then asks:

```
Name of file to copy to?
```

If you are copying the file on the same disk, type in a new, different file name and press [RETURN].

If you are copying to the other disk, type a file name (one that is not duplicated on the other disk) and press [RETURN].

Use the disk drive letter and colon (A: or B:) before the file name if you use the nonlogged disk drive for either "copy from" or "copy to."

WordStar makes a copy of the file.

If there is insufficient room on the disk for the copy, a message will say so. The file name used for the destination file will be left on the destination disk and should be erased. The original will not be damaged.

This copy file command offers a convenient way to transfer a file from one disk to another. First copy the file from your data disk to your WordStar disk. (This assumes there is enough room on the WordStar disk for the file — the WordStar disk doesn't have much room: see the section below on disk space.)

Then remove the data disk and replace it with another data disk. Now copy the file from the WordStar disk to the data disk, then use **Y** to erase the file from the WordStar disk.

Available Disk Space

The Osborne Executive's 5-1/4 inch disks can hold up to 185,000 characters of information.

When a disk is filled to capacity, you will get a "DISK FULL" error message while creating a document or while trying to edit a document.

It is vital to know how much file space is left on your data disks. Frequent "Disk Full" messages are a sign of inattention to your disk space.

WordStar has no file space program to determining how much room is left on a disk. There are, however, two programs on your WordStar disk that operate from the CP/M Plus operating system (outside of WordStar) and will give you different levels of information about your disk capacity.

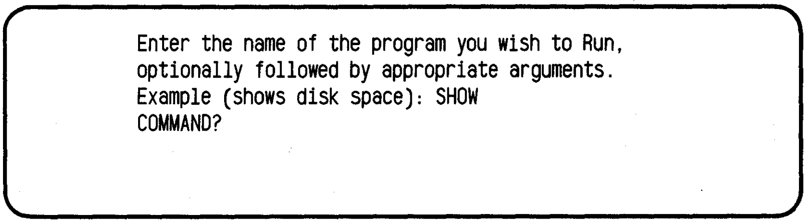
Knowing your disk space is so important to your use of WordStar that we will give you a step-by-step tutorial here on the use of the basic SHOW command.

[R][S][H][O][W] *Disk Capacity*

This program is run from the Opening Menu of WordStar.

1. Press R.

The screen clears and new prompts appear, including the following message:



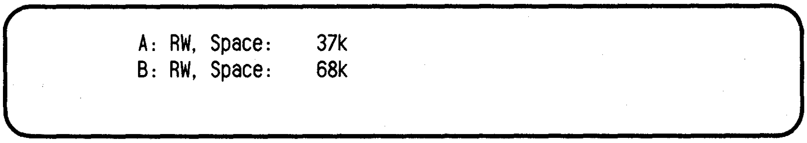
```
Enter the name of the program you wish to Run,
optionally followed by appropriate arguments.
Example (shows disk space): SHOW
COMMAND?
```

Ignore both the "explanation" and the example for now. The **R** command lets you run another program while still in the WordStar program.

2. Type the following:

[S][H][O][W][RETURN]

The screen will clear and after a moment, the SHOW program will display the disk space remaining on both your disks. Here is an example:



```
A: RW, Space:   37k
B: RW, Space:   68k
```

Hit any key to return to WordStar:

The first line tells you there is 37k, or 37,000 characters of space still remaining on the disk in drive **A.** The second line tells you there is 68k, or 68,000 characters, remaining on the disk in drive **B.** (A blank disk has 185,000 characters of space available.)

3. When you are finished examining this information, press any key on the keyboard. You will be returned to WordStar and the Opening Menu.
4. When interpreting the numbers, there are three parts to the disk storage system you must take into account in estimating how many more pages you can put on your disk.

   a. First, WordStar keeps a temporary file on the disk when you edit a long (10 page) document.

   b. Second, almost every file you edit will have a backup file. That means the space available must be divided in half.

   c. Third, a typical page of text uses about 2k of file space.

   Now let's combine these facts to decide in the example above how many pages of storage are available on the disk.

   a. Subtract 30k from the "Space: 68k" figure to account for WordStar editing space requirements.

      (If you habitually keep files more than 10 pages in size (NOT recommended), multiply the number of pages in your largest file document by 3 and use this number.)

   b. Divide this figure in half to account for backup files that will be automatically created whenever you edit.

c. Finally, divide the remainder in half again. The figure you get will be the number of pages of text your disk can still hold.

In the example above, you would subtract 30K from the 68K; then divide in half and divide in half again. The result is an estimate of slightly more than 10 additional pages of text before the disk is full.

While this may seem a complex undertaking, it is **crucial** that you examine your remaining disk space regularly, especially as your disk gets filled with files. A DISK FULL error message can force you to delete needed files and waste valuable time.

File Handling

Intermediate Level

DISK FULL Recovery Methods

Despite your best efforts you might get a DISK FULL message, usually when you try to save the file you have been editing.

It is possible to save your edited file by following this step-by-step guide.

There are two basic methods of recovering from a DISK FULL error message.

Method 1: Changing Disks

This method can only be used on SHORT documents, well under 10 pages in length.

1. Follow the DISK FULL instruction on the screen by pressing ESC. Then remove the data disk from drive **B**.
2. Replace it with a formatted data disk that you know has enough room.
3. Go to the beginning of the file and put ^KB to mark the beginning of a block.
4. Go to the end of the file (^QC) and put ^KK to mark the end of the block.
5. Enter ^KW, the Write To Disk command.
6. In answer to the prompt "Name of file to write to?", type in a file name. The block will be filed on the second disk under this name. You will be returned to your file in a moment.
7. When the file is finished, put the original disk back in drive **B**.
8. Quit the document with the ^KQ command.

Method 2: Erase an Old File

For longer documents, or when you want to keep the document on the same disk, erase some other file on the disk first. This involves several advanced commands which will be explained simply.

1. Follow the DISK FULL instruction on screen and press ESC.
2. Enter the command ^KF. The Directory of your logged drive will be displayed at the top of the screen.
3. Choose to delete a file(s) that will leave enough space to save your current file: a backup file, or erase another file that you have on a backup disk elsewhere.
4. Enter the command ^KJ, which erases a file while you are editing another file. In answer to the on-screen prompt, type in the name of the file to be erased. Repeat this step for each file to be erased.
5. Try again to save your document. If there is still insufficient room, delete more files.

File Handling

Advanced Level

Run a Program
Save and Exit to System
DIR
Exit to System
File Directory
Run SpellStar

Miscellaneous File Commands

R ***Run a Program***

From the Opening Menu, pressing "R" permits the running of any non-WordStar program residing on either disk.

Use the A: or B: prefix if the command is not on the logged drive.

Disks may be swapped in either drive (the WordStar disk may be swapped out until after the program is run, then returned).

When the program is completed or interrupted, or if a return to CP/M Plus is attempted, control is returned to WordStar. You may be prompted to "press any key to return to WordStar." If the WordStar disk has been removed, you will be prompted to re-insert it.

X ***Exit to System***

This command, issued from the Opening Menu, exits WordStar and returns control of the computer to the CP/M Plusoperating system. The **A>** or **B>** prompt will appear on the screen. (The screen will not be cleared.)

^K X *Save File and Exit to System*

This command is issued while editing a document. The file is saved to disk, and control is passed to the CP/M Plus operating system just as with the **X** command described above.

F *File Directory OFF/ON Toggle*

This command, issued from the Opening Menu, turns the file directory display OFF and back ON again.

Disk Space: DIR

DIR B: Disk Directory Command

The DIR.COM file on the CP/M Plus disk is used to provide a directory of files and the amount of space each uses. (This is not to be confused with the built-in CP/M Plus command DIR.) This command may be used to get a better idea of disk space availability than the SHOW command.

Locate the DIR.COM file on the CP/M Plus disk and use it to examine your WordStar disk. If there is sufficient room, use PIP (see *Volume 4 — Operating Systems*) to transfer the file to the WordStar disk. (DIR.COM may already be on the WordStar disk.)

Once this file is on the WordStar disk, it may be run from WordStar using the **R**, Run a Program, command from the Opening Menu.

Press R. Type in D I R and press RETURN to examine the logged drive (or add the disk drive letter and colon to examine the nonlogged drive).

WordStar automatically provides a full DIR listing (so you do not need to type in the [full] argument) of the drive. The listing gives file names, disk space consumed by each, and total disk space consumed. Subtract this total from 182k to determine disk space still available. (The disks hold 185k but the directory tracks take up 3k.)

DIR.COM shows all files, including system files.

If DIR.COM is deleted from the WordStar disk, entering the DIR command will give you the built-in CP/M Plus command DIR, which lists the files on the disk but gives no other information about the files.

Run SpellStar

S *Run SpellStar*

This is a special form of the "R = Run a Program" command described above.

This program, a spelling checker designed by Micropro, makers of WordStar, is not included on your WordStar disk but is available from your Osborne dealer. When obtained, the program interfaces with WordStar so this command may be used. See the instructions contained in the SpellStar user manual.

If SpellStar is obtained, issuing this command from the Opening Menu causes WordStar to search for and run the program SpellStar. As with the **R** command, when the program is finished you are returned to WordStar.

If SpellStar is not obtained, pressing this command will result in an error message. Press ESC to return to the Opening Menu.

Printing Files

Basic Level

Print a File

P ***Print a File (From Opening Menu)***
^K P ***Print a File (From Within a File)***

From the Opening Menu (not editing a file), press P to order printout of a file.

You are asked for the name of the file to be printed. Type a file name of a file on either disk (for the nonlogged disk drive, precede the file name with the drive letter and a colon, as A: or B:).

You can only print a file that is on the disk. So when you create a document, you must save it to disk using one of the Save File commands (^KD, ^KS, ^KX) before you can print it.

You can print a file while you are editing another file. The command for printing a file while editing is ^KP. (The command is shown on the ^K Block Menu.) As with the **P** command, you are asked for the file name. Type in the file name (preceded by the disk drive letter and a colon if the file is on the nonlogged disk drive).

With the ^KP command, you can print any file on either disk, including the most recently saved version of the file you are now editing. (You might want to review a draft of the file you are editing.)

Quick Tip:

PRINTING DRAFT COPIES WHILE EDITING

Since you can only print the disk copy of a document, not the version on screen, if you have made editing changes there will be discrepencies between the printout and the screen display. To get an accurate copy of the file you are editing, simply use the Save & Re-Edit command ^KS first. This will update the disk copy so it is identical to the version on the screen.

If you ask to print the disk copy of the file you are editing, WordStar says:

```
WARNING:  You are printing the same file as you are editing.
The last saved version will be printed, not reflecting un-
saved changes.  Furthermore, WordStar will not allow you to
save the file being edited while the print is in progress.
```

This is merely a reminder that what you will get is the disk version of the file, not the version on the screen, in case they are different. And since WordStar is using the disk file for printing, you can't enter a Save commands until printing is done.

Printing Options

ESC ***Bypass Printing Options***

When you have entered the correct file name of the file to be printed, press ESC. WordStar will then print your named file, from page 1 to the end of the file, obeying all printing and formatting commands that may be in the file. WordStar will assume you are using continuous-form paper and won't stop between pages.

There are options available permitting you to change any of these printing defaults (assumptions). To see the list of Printing Options, press RETURN instead of ESC after typing the file name.

You will then be given the chance to alter the defaults by selecting from a list of six Printing Options described below. When you have finished, printing will begin.

There are only three of these options that are usually of value at the Basic Level: Start at Page, Stop After Page, and Pause for Paper Change. We will describe these here.

For the other options, which will be listed in their proper order here but described in the Intermediate Level, press RETURN for each one to accept (not alter) the default of the option.

If you press RETURN through the entire list, WordStar will print the file using the defaults in the same way as if you had used ESC. ESC is simply a shortcut.

```
DISK FILE OUTPUT (Y/N):
```

Accept the default value of this option by pressing RETURN.

```
START AT PAGE NUMBER (RETURN for beginning):
```

If you press only RETURN without selecting a number, WordStar will start printing from the first page of the file. To start on any other page, enter a number and then press RETURN.

Pages are counted according to what number will actually be printed out on the page. If you altered the starting number by using the **.PN n** command, use the numbering series that command will introduce when telling WordStar to start printing on other than the first page.

When printing begins at the end of the Printing Options list, it may take a minute or two for WordStar to work its way through the disk file to the page you selected. The disk drive activity light will signal WordStar's activity.

```
STOP AFTER PAGE NUMBER (RETURN for end):
```

This command lets you stop printing on a given page. Enter the number of the last page you want actually printed, then press RETURN. As above, use the numbering sequence that will be printed on the pages.

For example, to print out only one page, page 5, in Start At Page you would enter the number 5 and RETURN. In Stop After Page you would also enter the number 5 and RETURN. WordStar would then print only page 5.

Press RETURN without entering any number and WordStar will print all the way to the end of the file.

```
USE FORM FEEDS (Y/N):
```

Accept the default value for this option by pressing RETURN.

```
SUPPRESS PAGE FORMATTING (Y/N):
```

Accept the default value for this option by pressing RETURN.

```
PAUSE FOR PAPER CHANGE BETWEEN PAGES (Y/N):
```

Use this option by entering Y if you do not have continuous form paper and must stop the printer to put a fresh sheet in after each page is printed.

WordStar will then stop after each page, with the message PRINTING PAUSED on the computer screen. Change the paper, then press P again (or ^KP if you are editing a file) to RESUME printing.

If you do not need to stop between pages, press RETURN.

Again, if you do not need any of these options, bypass them either by pressing ESC after the file name, or by pressing RETURN for each one of the options.

If you press the ESC key, printing will begin right away. If you went through the options list, there will be one final prompt:

```
READY PRINTER, PRESS RETURN:
```

When RETURN is pressed, printing will commence.

You can interrupt the printing command at any time before pressing RETURN in this final prompt, by using the command ^U.

Stop Printing

P Stop Printing (From Opening Menu)
^K P Stop Printing (While Editing)

If you are printing from the Opening Menu, once printing begins the Opening Menu is redisplayed with one change: "P = STOP PRINTING" is highlighted in place of "P=Print a document."

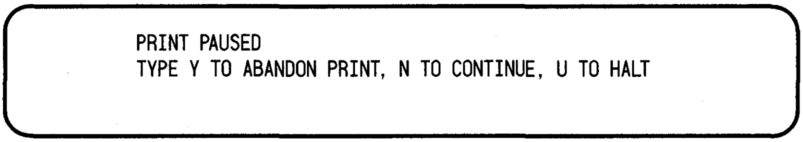
If you are printing while editing a file (^KP), the phrase "^KP=STOP PRINTING" is displayed in the menu area at the top of the page.

If you are editing a document, you can continue editing, but computer response time will be slowed. When the message "WAIT" appears on the screen, you will be obliged to pause.

(Special printer attachments called "print buffers" will receive and hold print file information during printout. They reduce the frequency with which the printer must go to the computer for data, and so allow editing to continue with fewer delays.)

To cause printout to stop, press P (from the Opening Menu) or ^K P (from within a document).

The following message will be displayed:



```
PRINT PAUSED
TYPE Y TO ABANDON PRINT, N TO CONTINUE, U TO HALT
```

Printing will continue until the printer buffer is emptied. (If you are using a print buffer as described above, the entire file may print out despite the Stop command because of the size of the buffer; it may be necessary to push the SELECT or PAUSE button on the computer.)

As the screen explains, when printing has been stopped, entering **Y** will cause WordStar to abandon printing completely.

Entering **N** will resume printout immediately.

If you must stop the printing in order to do something else at the computer, use **U** to temporarily suspend printing. WordStar will note where in the file printing stopped, and issuing another P or ^KP command will cause printout to begin again at that point.

Printing Test

WordStar provides codes for accessing a wide range of printer capabilities, from boldfacing to incremental spacing.

But not all printers can do all things. On your WordStar disk is a file called PRINT.TST that incorporates most of the available WordStar print features, and can be used to test your printer.

Print this file to see which of the WordStar commands your printer is able to obey. It will also give you an opportunity to make sure that your printer is properly installed.

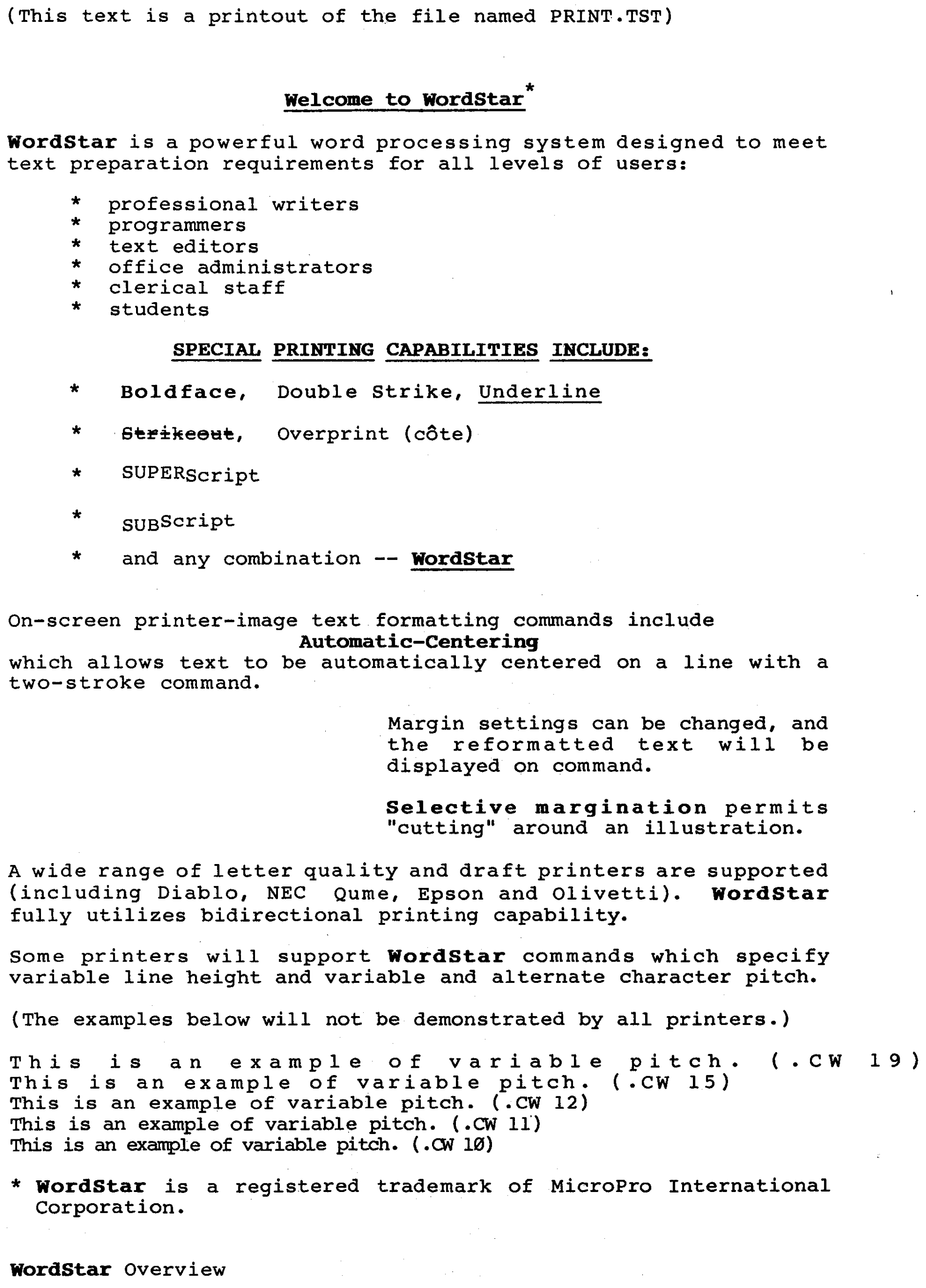
To print a copy of PRINT.TST, follow the instructions given above for printing: from the Opening Menu, press P, type the name P R I N T . T S T as the file to be printed, and press ESC to bypass the Printing Option list.

We've reproduced here what the file will look like if your printer has full print capabilities. Your printout may be different in any of the following areas:

- No overprint (co^te)
- Superscript and Subscript roll a full line instead of a partial line
- No variable pitch
- No variable line height
- No alternate pitch toggle ability

Having printed this sample, you have seen the range of your printer's capabilities and will know which ones you can use in the future.

And if you take a look at the PRINT.TST file with WordStar, you'll see some of the commands in this manual correctly applied in an actual document.



(This text is a printout of the file named PRINT.TST)

Welcome to WordStar*

WordStar is a powerful word processing system designed to meet text preparation requirements for all levels of users:

* professional writers
* programmers
* text editors
* office administrators
* clerical staff
* students

SPECIAL PRINTING CAPABILITIES INCLUDE:

* **Boldface,** Double Strike, Underline

* ~~Strikeout~~, Overprint (côte)

* $^{\text{SUPER}}$Script

* $_{\text{SUB}}$Script

* and any combination -- **WordStar**

On-screen printer-image text formatting commands include **Automatic-Centering** which allows text to be automatically centered on a line with a two-stroke command.

Margin settings can be changed, and the reformatted text will be displayed on command.

Selective margination permits "cutting" around an illustration.

A wide range of letter quality and draft printers are supported (including Diablo, NEC Qume, Epson and Olivetti). **WordStar** fully utilizes bidirectional printing capability.

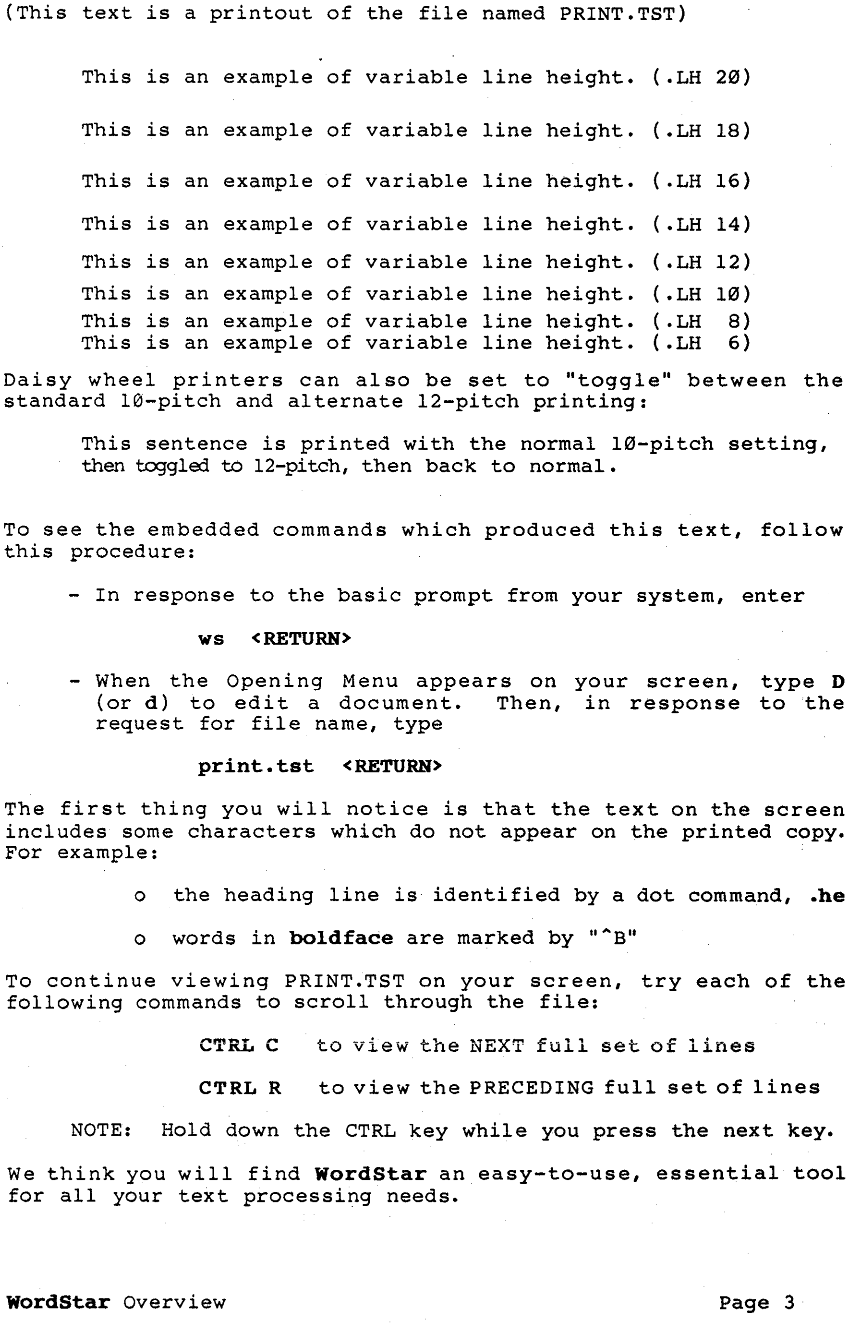
Some printers will support **WordStar** commands which specify variable line height and variable and alternate character pitch.

(The examples below will not be demonstrated by all printers.)

This is an example of variable pitch. (.CW 19)
This is an example of variable pitch. (.CW 15)
This is an example of variable pitch. (.CW 12)
This is an example of variable pitch. (.CW 11)
This is an example of variable pitch. (.CW 1Ø)

* **WordStar** is a registered trademark of MicroPro International Corporation.

WordStar Overview



(This text is a printout of the file named PRINT.TST)

This is an example of variable line height. (.LH 2Ø)

This is an example of variable line height. (.LH 18)

This is an example of variable line height. (.LH 16)

This is an example of variable line height. (.LH 14)

This is an example of variable line height. (.LH 12)

This is an example of variable line height. (.LH 1Ø)
This is an example of variable line height. (.LH 8)
This is an example of variable line height. (.LH 6)

Daisy wheel printers can also be set to "toggle" between the standard 1Ø-pitch and alternate 12-pitch printing:

This sentence is printed with the normal 1Ø-pitch setting, then toggled to 12-pitch, then back to normal.

To see the embedded commands which produced this text, follow this procedure:

- In response to the basic prompt from your system, enter

  ws <RETURN>

- When the Opening Menu appears on your screen, type **D** (or **d**) to edit a document. Then, in response to the request for file name, type

  print.tst <RETURN>

The first thing you will notice is that the text on the screen includes some characters which do not appear on the printed copy. For example:

o the heading line is identified by a dot command, **.he**

o words in **boldface** are marked by "^B"

To continue viewing PRINT.TST on your screen, try each of the following commands to scroll through the file:

CTRL C to view the NEXT full set of lines

CTRL R to view the PRECEDING full set of lines

NOTE: Hold down the CTRL key while you press the next key.

We think you will find **WordStar** an easy-to-use, essential tool for all your text processing needs.

WordStar Overview Page 3

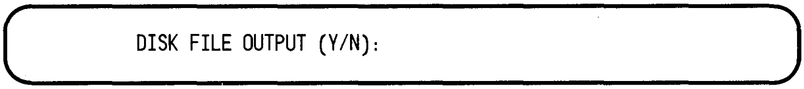
Printing Files

Intermediate Level

More Printing Options
Chaining Print Files
Multiple Copies

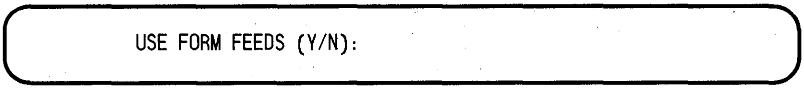
More Printing Options

This section describes in more detail the other Printing Options listed in the Basic Level.



```
DISK FILE OUTPUT (Y/N):
```

If you select yes, WordStar will make a separate copy of the file on the disk instead of sending it to the printer. This is a specialized option which is rarely selected since the disk file is modified and formatting dot commands are not functional in it, and copies of the disk files can more easily be made with the O, COPY, command. Press RETURN to skip.

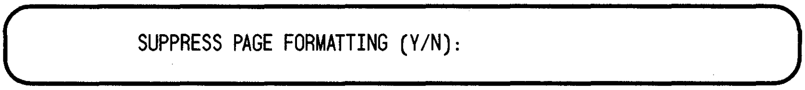


```
USE FORM FEEDS (Y/N):
```

The default on this option (N), which is selected by pressing RETURN, lets WordStar tell the printer how long each page is. WordStar normally sends a series of RETURNs to skip the printer down to the top of the next page.

But some printers are able to keep track of their own page lengths and skip to the top of the next page. If your paper lengths vary often, this command lets you use the WordStar default page length of 11 inches (66 lines) at times, and use the printer's switch-set page length at other times.

To let the printer do the form feed, enter a Y to this option. WordStar will then send the printer a "form feed" character (OC hex in computer language) and the printer will take care of moving to the next page. Not all printers are able to do this.



```
SUPPRESS PAGE FORMATTING (Y/N):
```

Normally the answer is No, which you can select by entering an N or by simply pressing RETURN.

If you select Y, WordStar will print on the page all the dot commands used in the file. The dot commands will be printed but not obeyed, so the formatting will not occur on the printout.

This option is used to document the dot commands used in a document.

Multiple Copies

MailMerge, a program included on the WordStar disk, produces mailing lists and form letters, and is detailed in the next section of this manual. MailMerge also has extra printing capabilities that are valuable for everyday print output. We will now examine two of these: Print Multiple Copies and Chain (Insert) Files.

M *Merge Print (Opening Menu)*

The P and ^KP print commands produce one copy. For more than one copy, use the **M** print command, which activates **MailMerge**'s print routine.

The **M** print command is used only from the Opening Menu (it cannot be activated from within a document). Instead of pressing P to print, press the letter M for **Merge-Print**.

As with the P command, you will be prompted for the name of a file to "merge-print."

Type the name of the file, then press RETURN (DON'T use the ESC key to bypass the options list — the multiple copies option is in that list).

The same options list that you would see for the P = Print command appears, except that there is an additional option that follows "Stop After Page Number:"

```
NUMBER OF COPIES (RETURN for 1)?
```

Enter a digit for the number of copies desired, and press RETURN.

Continue through the remainder of the options list in the same way as with the Printing Options list, pressing RETURN to skip options you do not wish to change.

The number of copies you chose will be printed. The same Stop Print and other commands operate with Merge-Print as with Print.

Chaining Print Files

. F I *File Insert (MailMerge Dot Command)*

It's sometimes convenient to break a long document into several shorter files. For final printout, however, the files must be merged together.

"Chaining" is a convenient way of linking files only during printout, so that the printout appears to be of one long file even though the actual files remain separate on the disk.

Chaining is accomplished with the .FI "file insert" MailMerge dot command and the M Merge = Print command.

Here is how to chain your files:

1. When you are ready to print, use the D = Edit a Document to enter the first of the files you want to be printed. Go to the end of the file.
2. On the last line of the file, insert the following:

   .FI filename

   In place of "filename," type the full name of the second file to be printed.
3. Save this first file.
4. From the Opening Menu, press M for Merge-Print. Type the name of the **first** file. Proceed through the print options list in the usual way.
5. The first file will be printed. When Merge-Print reaches the end of the first file, the ".FI" command tells it to "insert" the file named there.

   Merge-Print goes to the named file and prints it immediately following the first file. On paper they will both appear to be one long file with no evidence of a break.
6. You may chain as many files as you wish in this manner. Simply enter each file and add at its final line the ".FI" command and the name of the next file to be printed. Save each file, repeating this procedure for each file except the last one.

   In this way, a long document may be printed as one unit although maintained as separate disk files for editing and filing convenience.

7. The .FI command will only be active with the Merge-Print printing command. The P = Print command will ignore any .FI commands and print only one file.

   For this reason, the P = Print command can be used to print just one file out of the sequence without having to remove the .FI command from the end of the file.

8. With Merge-Print, each of the files is printed starting where the previous file left off on the page. It will not start a new page. If you want a file to begin on a new page, add the .PA, Page Break, command at the beginning of that file.

Printing Files

Advanced Level

Advanced File Chaining

More on File Chaining

Merge-Print treats a series of files linked by a File Insert command exactly as if it were one long file. Any toggles switched ON in one file will remain ON through all succeeding files unless switched OFF. Thus a heading initiated in one file will appear in all subsequent inserted files unless switched off.

This fact suggests a more convenient method of printing a series of files. If file names, file sequence, headings or other factors are being changed frequently, it can be time-consuming to go into each file to change the .FI command over and over.

An easier way to accomplish the same end is to create a PRINT FILE.

1. Create a file named PRINT.FIL. This file will consist of:

   a. a series of File Insert (.FI) commands on separate lines, listing in order the files to be printed;

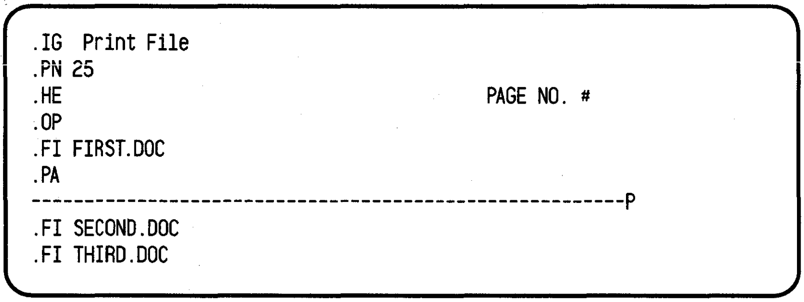
   b. whatever page break (.PA) commands may be required, inserted between the file insert commands;

   c. whatever heading, footing and format instructions apply to the entire document;

To print, use Merge-Print to call up "PRINT.FIL." It will act as a command file and chain the entire series of files, with headings and page break separations where needed.

If files are merged, document names changed, or only some of the files need to be printed, the "PRINT.FIL" document can be quickly and easily edited to reflect those changes.

The following is an example of a PRINT.FIL chaining three documents: FIRST.DOC, SECOND.DOC (which starts on a new page) and THIRD.DOC (which continues on the same page as the end of SECOND.DOC). The phrase "PAGE NO. #" will appear at the top of each page, the numbering starting with 25.



```
.IG  Print File
.PN 25
.HE                                      PAGE NO. #
.OP
.FI FIRST.DOC
.PA
-----------------------------------------------------P
.FI SECOND.DOC
.FI THIRD.DOC
```

PART 2
MailMerge

Introduction

MailMerge is a printing program that, in combination with WordStar, prints individual form letters and repetitive documents by combining variable information, such as names and addresses, with standard information, such as the body of a letter.

You need only type the standard information once, into a WordStar file. The variable information goes in a second WordStar file. MailMerge then combines the two files during printout to produce the series of "personalized" letters.

MailMerge will even print the mailing labels for the letters.

MailMerge has many other powers that reduce repetitive typing.

MailMerge operates during **printout**. Using the M=Merge-Print command instead of P=Print a File causes MailMerge to obey special dot commands inserted in the files.

Form Letters

Basic Level

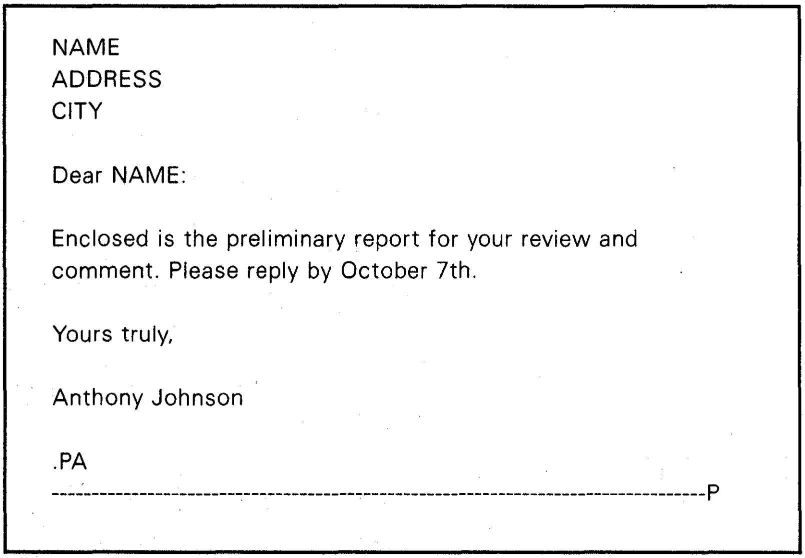
Form Letter

The typical MailMerge document consists of at least two WordStar files. One is the master form letter, which contains the parts of the basic letter that do not change. The other is the data file, which contains all the variables.

To demonstrate how MailMerge works, we'll walk through a sample form letter and data file which use the most important MailMerge commands.

To write the form letter which follows, start WordStar and open a Document File with the D command. Give it the name FORMLETT.

Now type the following letter.



```
NAME
ADDRESS
CITY

Dear NAME:

Enclosed is the preliminary report for your review and
comment. Please reply by October 7th.

Yours truly,

Anthony Johnson

.PA
-------------------------------------------------------------------------------P
```

This is a form letter which will be sent to many different people. The name and address on each letter will change, but the body of the letter will remain the same.

To indicate where these **variables** will go, you have typed NAME, ADDRESS and CITY.

The .PA (Forced Page Break) command at the end ensures that each letter will print out on a separate sheet.

The standard parts of the form letter are now complete. Make whatever corrections are needed and reformat (^B) the letter.

[&] *Keyword Indicator*

Now you can enter the commands that make this a MailMerge document. This involves marking off the variables so MailMerge can find them. Do this by putting an ampersand **&** before and after each variable. The top part of you letter will then look like this:

```
&NAME&
&ADDRESS&
&CITY&

Dear &NAME&:
```

The & (or "ampersand") shows MailMerge the name and location of each KEYWORD, or variable, in the form letter. The keyword itself is the word enclosed in the ampersands. There are three keywords in this letter: NAME, which is used twice, ADDRESS and CITY.

Once you have added the ampersands, SAVE the document with ^K D.

The Data File

Now let's create the Data File that will contain the actual names, addresses and cities to be used in the form letter.

Use the N command to open a Non-Document file and call it FORMDATA.

Remember, this is a Data File, not a Document File, so use the N = Open a Non-Document file command.

For this example, enter the three addressees given below. Because Wordwrap is OFF when you are in Non-Document mode, you must put the RETURN at the end of each line.

```
George Drummond,65 Kelly St.,San Francisco CA 94580
Alice Farnsworth,236 Green Street,Atlanta GA 30232
Bill Gibbons,166 N. Real Ave.,New York NY 10015
```

Make whatever corrections may be needed, noting that in the above sample there is NO SPACE following the commas, and there is NO COMMA between the city name and the state. Edit your sample list until it is the same way.

MailMerge will be able to read this file and put each name, address and city in the proper place in each form letter.

The **comma** tells MailMerge where one variable in the data file ends and the next one begins. To MailMerge, a comma is the "delimiter," or separator of variables. The RETURN at the end of each line is another form of delimiter marking the end of a variable.

That's why you don't need a space between "Drummond" and "65;" because of the comma, MailMerge will stop after "Drummond" and go back to the form letter. The address is already separated in the form letter because it is on the next line.

That is also why you put no comma between city and state: MailMerge would think the state was a new variable, and we don't have a keyword for that in the form letter.

A Rule:

> *The variable items—from one comma to the next—can be up to but not more than 200 characters.*

Now SAVE the data file with ^KD.

Form Letter Again

Two files have been written: a form letter with the keywords indicating the location of the required variables, and a data file with the actual variables in it.

Now you need to connect the two files with special dot commands so MailMerge will know which data file to use and how to use it.

. D F *Data File*

Use D to edit the file FORMLETT in the Document mode. (The form letter is prepared in Document mode like an ordinary letter; only the data file, with its variables, must be done in Non-Document mode.)

Insert as the very top line of the file, above &NAME&, the following:

```
.DF FORMDATA
```

This gives the name of the data file ("DF") that contains the variable data to be used by this form letter.

. R V *Read Variables*

Now insert this line between .DF FORMDATA and &NAME&:

```
.RV NAME,STREET,CITY
```

This line lists the keywords from the form letter, telling MailMerge the exact order it will find the variable information that goes with these keywords IN THE DATA FILE (not in the form letter).

Each line of our data file contains a Name, a Street, and a City, in that order. (Although it also includes the state and ZIP code, we're calling it City for short.) So the .RV line lists the corresponding keywords in that same order: NAME,ADDRESS,CITY.

When MailMerge prints the file, it will know which Data File to get the variables from and in what order the variables will be in that file.

Rules:

The keywords in the letter and in he .RV line can be up to 40 characters long, must begin with a letter, and can include numbers and hyphens, but no punctuation marks.

The .DF and .RV commands always go together, .DF first and before any text in the form letter; .RV after .DF and also above all text. There can only be one .DF. If the .RV line is too large to fit within the right margin, it can be continued on the next line by repeating the .RV command; MailMerge will treat them all as one big line during printout.

The form letter is now completed. SAVE it with the ^K D command.

Merge-Printing

M *Merge-Print*

Your sample MailMerge file is ready to print!

From the Opening Menu, press M, the Merge-Print command. Merge-Print calls up the MailMerge program.

Type the name of the form letter file: F O R M L E T T.

Merge-Print has a series of print options just like P=Print a File. You can press ESC to skip the options list, or use the options list as you would with the P=Print a File command. You will find only one additional option: "Number of Copies?" for printing more than one copy. Skip this option for now by pressing RETURN.

When the printer begins, you have only a moment to wait before seeing three form letters with a different name and address on each one.

How did MailMerge do it?

The first keyword in the form letter, &NAME&, sent MailMerge to the data file named in the .DF line. MailMerge looked for a variable with the keyword NAME. The .RV line told MailMerge that NAME would be the **first** item in the line.

So MailMerge took that item (everything up to the first comma) and used it to replace the keyword &NAME& in the form letter.

The second keyword, &STREET&, according to the .RV line, is the second item in the data file. The third keyword, &CITY&, is the third phrase.

The keyword &NAME& in "Dear &NAME&" was simply a repeat of the earlier identical keyword &NAME&. If any of the other keywords (&STREET& or &CITY&) had been repeated elsewhere in the letter, MailMerge would have repeated them in the same way.

Refining the Form Letter

"Dear George Drummond" is awkward; preferable might be "Dear George." Let's modify the form letter and the data file to make this improvement.

Access the form letter file using D=Document mode.

To break "George Drummond" into two parts, we must break the keyword &NAME& into two keywords. Delete &NAME& and enter &FIRST& and &LAST& in its place.

Now delete &NAME& from "Dear &NAME&." Substitute "Dear &FIRST&." This will cause "Dear George" to print instead of "Dear George Drummond."

Go to the top of the page and delete NAME from the .RV line and replace it with FIRST,LAST, to reflect these changes.

.OP *and* .PN *Page Numbering*

Because MailMerge numbers pages consecutively just as Print does, your three sample letters came out numbered 1, 2 and 3. For these one-page letters, fix this problem by adding the dot command .OP, Omit Page Numbering. If you were printing a letter of more than one page, you might want the pages numbered: then use the page numbering dot command .PN. It can be put at the top of the page as .PN 1 to ensure that each letter begins numbering with 1. Or to number starting with the second page, put .OP at the top of the first page (to prevent the first page from being numbered) and place .PN 2 toward the bottom of the first page.

SAVE the form letter file with ^KD.

Now edit the data file FORMDATA using the N=Non-Document command.

This involves simply putting a comma between the first and last name of each person in the file, and deleting the space following each comma. MailMerge will now consider each name to be two variables.

Save the data file with ^KD.

Now print the form letter using M=Merge-Print and see the results.

From this short exercise you have seen the essential MailMerge in action. The remainder of this section offers refinements, additions, and new applications of the principles you have learned so far.

Form Letters

Intermediate Level

Ask Value (.AV)
Optional Keyword
Empty Variables
Reformatting
Set Value (.SV)
Display Message (.DM)
Chaining Files (.FI)
Multiple Copies

Form Letter

.AV ***keyword*** ***Ask Variable***

The basic form letter has been demonstrated. Now let's consider some refinements.

The date of the sample form letter will change each day the form letter is printed. This introduces a variable that must be handled in a slightly different way.

The date could be written into the sample letter and edited by WordStar every time it changes. Or it could be entered into every data line in the data file. Another way to handle such a variable is the .AV dot command, which asks for the variable on screen when needed.

To demonstrate this command, edit the form letter (use D=Document mode) and insert at the top, following the .RV line, the following:

. A V SPACE BAR D A T E

Then insert the keyword & D A T E & above the body of the letter where you would normally put the date of a letter.

Save the file, then print to see the Ask Variable command in action.

When you print, the screen of the Osborne Executive says:

```
DATE?
```

You respond by typing in the date. When you press RETURN, the date is printed on the letter, and the rest of the printing proceeds as before.

You will be asked for the date as each letter prints.

You may use as many .AV commands in the letter as you wish, and they can be anywhere in the text or above the text as long as they precede the line containing the keyword referenced. Each .AV command is followed by a single keyword. (As with all dot commands, the Ask Variable command must be on a line by itself.)

. S V *Set Variable*

Another way to handle the problem posed by the date is to set the value of this keyword variable once each time you print.

To accomplish this, insert in the command section you are building at the top of the page (the area with all the MailMerge dot commands) the following:

```
.SV DATE, March 29, 1983
```

This takes the place of the .AV command. Both commands can be used in the same form letter as long as they affect different keywords. Both commands can be used more than once in the same form letter, as long as each command is on a separate line. Like .AV, .SV can be located anywhere in the document before the keyword it affects.

This command can be used for a keyword that is used in the form letter several times; when the value of the keyword changes, simply edit the one line at the top of the page.

. D M ***text*** ***Display Message***

This command can be used to explain the Ask Variable request in more detail; several lines of messages can be displayed and all will remain on the screen until scrolled up by more messages, or until the Clear Screen command (described below) is used.

Use .DM to make clear what information is needed to answer an Ask Variable request. You can put keywords in the Display Message to customize the exact message displayed on the screen. Here is an example:

```
.RV  FIRST,LAST,ADDRESS,CITY,STATE
.DM  Letter to &LAST& of &CITY&.
.AV  DATE
```

MailMerge will insert the current value of the LAST name and of the CITY and display them on the screen. On the next line it will ask DATE?. You will now know which letter is being printed and can backdate or postdate the individual letter.

This is particularly useful for inserting specific dollars and specific names in a document: the message displayed can be customized to show you what is being printed so the correct entry can be made from the keyboard.

For the message to display properly when keywords are used, the .DM command must follow the .RV command and precede the .AV command.

Notice that the keywords in the .DM line are enclosed in ampersands (&). When the keyword is enclosed in ampersands, it tells MailMerge to insert the value of the keyword (the actual variable) in that place. Without the ampersands, the keyword is part of the command, directing MailMerge to a specific activity, in order to obtain a value for the keyword.

. C S *Clear Screen*

The screen can get crowded with messages being displayed, especially since the messages don't replace one another, they are simply added to the bottom of the list. The Clear Screen command preceding a Display Message or an Ask Variable command will clean things up by erasing whatever is present on the screen. You can add a message on the same line and it will be displayed after the screen is cleared of prior messages.

& ***keyword*** / O & *Optional Keyword*

Some addressees have more than one line in their address: those with suites, foreign addresses, and corporate names all add an extra line to the address block:

Raymond Vassey
243 Forest Lane
Suite 1013
Sherman Oaks CA 90405

It is possible to add a second address line, ADDRESS2. But if there is no value to put in that line, it remains blank:

Raymond Vassey
243 Forest Lane

Sherman Oaks CA 90405

The /O (slash, capital O) command, when attached to a keyword, tells MailMerge that if there is no entry for this keyword, just skip it — don't leave a blank line.

In the second example above, the entry **&ADDRESS2/O&** in the form letter would have resulted in the CITY line being printed directly under the STREET line, instead of an extra line below it.

The /O command goes with the keyword surrounded by ampersands in the body of the letter; it is not used in the .RV or .AV lines.

Empty Variables

In the data file, there will sometimes be no variable corresponding to a keyword listed in the .RV line of the form letter. For example, while two lines might be allowed for the address (ADDRESS1 and ADDRESS2), some addressees will have only a one-line address. This requires leaving a variable value blank.

When you come to an empty variable like this, you must put in the comma even though there is no data for that field.

MailMerge finds the variables to put into the keywords by counting over from the left margin in the data file, counting each comma as marking the end of one variable. If a field is left out, the comma must be left behind to indicate an "empty" variable. Otherwise MailMerge will count over and pick up the CITY and put it in the second address line (in the example described above); then put the name from the next data line in the CITY location of the form letter, and so on, throwing the entire series of letters out of synchronization.

Here is an example of what the data line will look like if there are two address lines but the letter goes to someone with only one address line:

Raymond Vassey,243 Forest Lane,,Sherman Oaks CA 90405

MailMerge will go to the third item and find it blank: there is no character between the second and third commas. MailMerge will leave it blank, as it should.

Always put the proper number of commas for the variables, even if some are blank.

Partial Files

You may have two different form letters that each use part of the same data file for the basic information. For example, one form letter may use the name and address from a data file, another may use the name and account number from the same data file.

The data file must contain all the information required by both form letters so there is data for each keyword in both form letter files. It doesn't matter that the data file contains extra variables not required by one or the other of the form letters. In other words, a form letter may use a subset of a data file's variables.

For instance, your data file may list an account number:

Raymond Vassie, 243 Forest Lane,,Sherman Oaks CA 90405,
A30407-83

One form letter uses that account number, another does not. In the form letter that doesn't use the account number, you must still list a keyword for the data file's account number in the corresponding order on the .RV line at the top of the form letter. This is so MailMerge can account for each item in the data file, even though not all items are used. The keyword standing for the account number is simply not used as a keyword in the body of the letter and is therefore never called for.

Nor must the data file's variables be listed in any particular order with reference to the order in which the keywords are called for in the form letter: as long as the Read Variable (.RV) line accurately reflects the order of the data file's entries.

File Insert

. F I ***filename*** ***File Insert***

In addition to calling a data file, MailMerge can insert one WordStar document file into another.

Put into a file the dot command .FI, followed by the name of another WordStar document file, for example:

.FI MAY23.LTR

and during Merge-Print, MailMerge will go to the second file at that point and start printing the file named MAY23.LTR from the same disk.

After printing the second file, MailMerge will return to the first file and continue printing where it left off.

This allows you to insert a "boilerplate" or standardized paragraph in the middle of a letter without having to copy or type it into the file.

You can put as many of these File Insert commands into the main document as desired. You can also put a File Insert command into the file being called, and another File Insert command into that file, and so on with each file calling a subsequent file. However, there is a limit of seven levels of inserts.

File Insert allows you also to chain a number of documents one after another, so that a single long document broken up into small files can be linked together into one long document at print time, rather than physically combining them into one long WordStar file. Simply put the .FI command on the last line of each document, with the name of the next file. (Each File Insert command must end in a RETURN.)

Because this is equivalent to each file calling the next file, only seven files in a row can be called in this way, as mentioned above. To get around this limitation, create a Print File containing all the File Insert commands.

A Print File is created by opening a document file in WordStar. Basically this file will consist of nothing but a list of File Insert commands, thus:

```
.FI Chapter1.doc
.FI Chapter2.doc
.FI Chapter3.doc
```

Since each called file returns to the Print File when completed, as many files can be linked in this way as you wish.

The Print File can contain other print formatting commands controlling the entire final document. Here for example is a print file which controls the headings, page divisions, and page numbering of the entire document being linked:

```
.IG draft of 27June
.PN 45
.HE Great American Novel                            Pg. #
.OP
.FI Chapter4.doc
.PA
-----------------------------------------------------------P
.FI Chapter5.doc
.PA
-----------------------------------------------------------P
.FI Chapt6A.doc
.FI Chapt6B.doc

"To Be Continued!"
```

The final comment, "To Be Continued!", will print out two lines below the last line of file Chapt6B.doc.

This same strategy can be used to print a document consisting of a series of boilerplate paragraphs, each paragraph in a separate file. The following Print File:

```
.IG contract for Lewis Engineering
.FI DEC27.DOC
.FI para27
.FI para3
.FI para14
.FI sigblk2
```

would print a document consisting of an introductory paragraph (DEC 27.DOC), followed by three paragraphs from a numbered list of boilerplates (para27, para3, para14), and signature block number 2 (sigblk2).

Thus MailMerge can be used to construct elaborate documents of standard components printed in any desired order.

Reformatting

During printout, MailMerge automatically reformats text lines to account for variables so they conform to the existing margins and tabs of the form letter.

You may use the Conditional Paging (.CP) dot command just as in other WordStar documents to ensure that this automatic print-time reformatting does not cause dangling heads and widow lines.

Multiple Copies

MailMerge makes a single copy of each unique letter until it runs out of data lines in the data file.

You can make duplicate copies of each letter. As with the P=Print command, press RETURN after the M=Merge-Print command to view the options. Merge-Print has one additional option not present in P=Print: "Number of Copies?" Enter the number of duplicates desired and that number will be printed of each unique letter.

Form Letters

Advanced Level

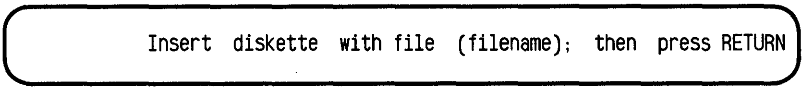
CHANGE disks
Reformatting (.PF)
Error Messages
Repeat Printing (.RP)
Justify (.IJ, .OJ)

CHANGE Disks

You can combine data files and form letters that are on different disks, or File Insert files on different disks, by using the CHANGE command.

In the .DF Data File line of your form letter, after the name of the data file to be used, put a comma and the word CHANGE (all capitals).

This tells MailMerge the data file named in that line is on another disk. During printout, when the data file is required, MailMerge puts this message on screen:



```
Insert diskette with file (filename); then press RETURN
```

The name on the .DF line will be inserted. Change disks and press RETURN.

BE SURE that the disk that must be changed is not the WordStar disk: MailMerge needs this disk during printing and will ignore a request to take out the WordStar disk from drive A.

If you are using the File Insert command .FI, follow the file name by a comma and the word CHANGE if the file being inserted is on another disk. Again you will be prompted to insert the disk with that file, and then to switch back when the file is done.

Repeat Printing

. R P n *Repeat Printing*

This is an alternative to using the Merge-Print option "Number of Copies" to get duplicates.

To get more than one copy of each document being merge-printed, insert the dot command .RP followed by the number desired.

Typically this command is put at the beginning of the form letter or form document file along with the other MailMerge dot commands. It may be put at the end if necessary but must be followed by a RETURN.

This command will only work in a file that has the .RV Read Variable in the same file and uses a datafile. It will be ignored otherwise.

MailMerge will make the copies by running through the entire form letter/data file sequence the required number of times. If you have ten letters being printed with two copies of each, MailMerge will print all ten letters once, then print all ten letters a second time, rather than print each of the letters twice before printing the next letter.

Print Formatting

MailMerge automatically adjusts line lengths in lines that have variable information, so they conform to the form letter's margins and tabs, and to the form letter's right justified or ragged right margin.

There are situations in which you will not want MailMerge to make this adjustment, or you will want a different adjustment made.

.PF OFF
.PF ON
.PF DIS

Print Time Formatting

To stop MailMerge from formatting a line or section of text, enter .PF OFF as a dot command on a line by itself immediately above the section in question. Starting at that point, reformatting will not take place. The lines will be printed as they are, including any addition of variables that change line endings. This is useful for data lines, names and addresses, and telephone numbers that might be split onto a new line.

You can end .PF OFF by putting .PF ON at the end of the section.

You can precede boilerplate paragraphs intended for File Insertion with .PF OFF or .PF ON to control formatting. At the end of those files, however, use .PF DIS to turn off the format command issued, rather than .PF OFF or ON.

DIS cancels the immediately preceding .PF command and returns MailMerge to whatever format it was using, whether ON or OFF. It cancels the previous format command.

This allows a return to the previous formatting status without the need to know what that status was. This is particularly useful in file inserts.

. O J SPACE BAR O N ***Output Justification***
. O J SPACE BAR O F F
. O J SPACE BAR D I S

. I J SPACE BAR O N ***Input Justification***
. I J SPACE BAR O F F
. I J SPACE BAR D I S

When print time formatting is in effect, lines are reformatted to conform to the existing margins; they are also made to conform to the existing text's right justified or left ragged right margins.

The Output Justification commands control how the final printing will look.

.OJ OFF prints out everything in ragged right.

.OJ ON prints out everything right justified.

.OJ DIS lets the document print out whatever way it already is; any variables will be reformatted to conform to the status of the existing text, whether ragged right or right justified.

Individual paragraphs can be bracketed with these commands to customize output.

During MailMerge, the various documents being printed may be in different justification status; some may be justified, others ragged right. Ordinarily, MailMerge determines whether a paragraph is justified by seeing if its lines conform to the right margin: there is no "code" embedded in the text.

It is possible that a paragraph may not have an even right margin, yet you wish it to be considered justified for the sake of later print output control. You can precede such a paragraph by the command .IJ ON; this will cause MailMerge to consider the following text (until otherwise noted) to be right justified, regardless of how the text actually looks.

The .IJ OFF command does the opposite: tells MailMerge to consider the following text to be ragged right, even if it looks otherwise.

.IJ DIS tells MailMerge to go back to considering the text on its merits: right justified if it is even with the right margin, ragged right otherwise.

Various combinations of the Input Justification and Output Justification will allow you to customize documents which you build from complex combinations of data files and file inserts. It gives you maximum control over the results. Experimentation will suggest other uses.

All of these formatting and justifying commands can be used wherever they apply in a document, data file, or boilerplate file. As with all dot commands, each must be on a separate line of its own, beginning at Column 1.

Error Messages

MailMerge only displays error messages during printout. Most are self-evident, but a few require specific interpretation:

```
***Invalid Dot Command ignored:
```

An incomplete or irrelevant dot command will be bypassed and printing will continue as best it can without the dot command. The invalid dot command will be printed on the screen, although its location will not be given. You can check the file and watch the flag column for a question mark, which indicates a dot command that cannot be understood by MailMerge or WordStar.

```
***WARNING:  Overlong data value truncated.
```

One of the variable items in the data file is more than 200 characters long. MailMerge will use the first 200 characters and ignore the rest. This message often occurs if a comma was accidentally deleted or a close quotes omitted from the data file line, resulting in an overlong variable.

```
***Invalid variable name in .RV command ignored.
```

The variable name was longer than 40 characters, or it did not start with an alphabetic character as the first character, or it included a punctuation mark other than a hyphen—all of these are illegal.

```
***WARNING:  Data exhausted, null value(s) used.
```

This usually occurs due to a mistake in the data file. If one data line had the wrong number of variables in it, had a comma left out or a close quotes (") omitted, then the number of elements in that data line will not conform to the .RV line in the form letter.

MailMerge will then use the next variable, throwing the variable sequence off, and putting variables in the wrong places. At the end of the file, MailMerge will come up one or more variables short. This message results.

Check your printer output to determine where the error occurred. You will find names in address locations, cities in names, and other mismatches.

This message may also occur when printing labels three across if the final print line has only one or two labels in it. The message may then be ignored.

APPENDIX 1
Modifying WordStar With INSTALL

Install

Basic Level

Printer Installation	*Preparing WordStar*
Selecting Printers	*Protocol*
Printer Driver	*Quitting*

Word processing programs must be "installed" to match the operating requirements of the computer and the printer.

WordStar has already been installed for the Osborne Executive and is ready to use without further modification.

Printer Installation

There are a variety of printers available, using a variety of methods of communication. The Install program allows you to customize WordStar to match your exact printer type.

WordStar has already been installed to operate a standard Centronics (parallel) printer similar to an Okidata 82 or an Epson MX-80 dotmatrix printer (without Graftrax).

If your printer is properly set up and the cable connecting it to your Osborne Executive is correctly wired, you should be able to print your documents without further adjustment.

Test that your printer is properly installed by printing the test document PRINT.TST that is on your WordStar disk. Do this by starting WordStar, then pressing P for Print; then type in the name P R I N T . T S T and press RETURN.

IT IS STRONGLY RECOMMENDED that your printer be installed and connected by the vendor, and that the vendor demonstrate to you that the printer works with your WordStar, before you take delivery of the printer. Printer installation and cabling can be a complex job.

If your printer works properly, you should have no need to use the Install program described below until you have reached the Intermediate Level.

If your printer is not a Centronics style printer, then you must use the INSTALL program to adjust WordStar to the printer.

The installation process consists of two steps:

1. Preparing WordStar for installation
2. Choosing among the menus offered:
   a. Installing terminal type
   b. Installing printer type
   c. Installing selected WordStar defaults

In this section we will explain how to prepare WordStar for installation and how to select a printer type.

Selecting WordStar defaults will be discussed in the Intermediate Level, and terminal types in the Advanced Level.

If you must install your printer or cable, we suggest you also review Appendix 2, Connecting a Printer, in this volume.

Preparing WordStar

The INSTALL program itself is too lengthy to be put on the WordStar disk: you will find it on a separate disk of its own. The disk has two files: WINSTALL.COM and WS.INS.

AFTER you have made working copies of your master disks, put the working copy of your **INSTALL** disk in drive **A.**

Put the working copy of your **WordStar** disk in drive **B**.

Press RETURN. The INSTALL program will automatically load. When it is finished the first screen of Install will appear to explain what you should do.

The Install program is carefully designed to explain every step as it proceeds. You will have to read several screens of information and follow the directions to begin the installation process.

On one of these screens, you will be asked to enter the drive letter for the WordStar files.

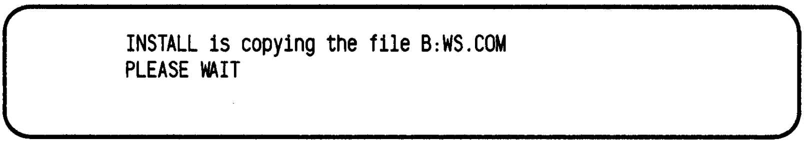
Enter B : RETURN.

You will be asked to confirm that you are re-installing the file B:WS.COM. Press RETURN to confirm.

This confirmation is requested several times. Follow the instructions given on the screen.

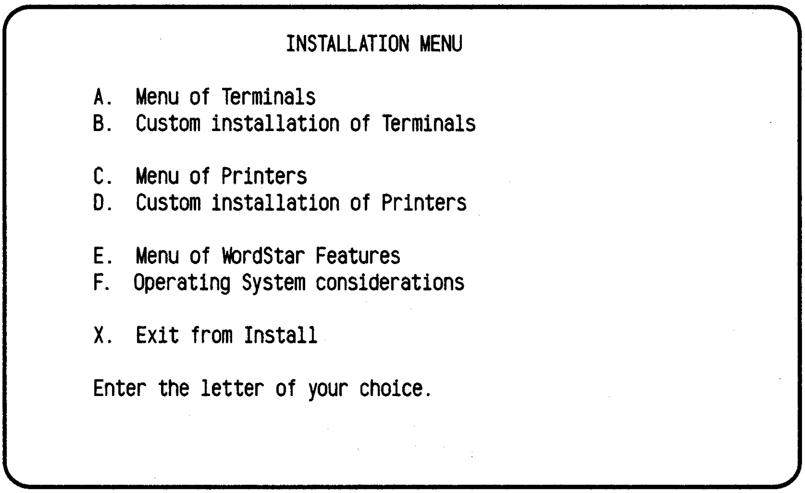
You will be warned that the new WordStar will overwrite the old WordStar. Agree by pressing RETURN. (You will have the opportunity to back out of the Installation if you change your mind later.)

Finally, the following message appears:



```
INSTALL is copying the file B:WS.COM
PLEASE WAIT
```

It takes a moment for the copy to be made. Then the INSTALLATION MENU appears, offering you your options. After you select each kind of option, you are returned to this Installation Menu, also called the Main Menu, until you are finished installing. The Installation Menu looks like this:



```
                    INSTALLATION MENU

    A.  Menu of Terminals
    B.  Custom installation of Terminals

    C.  Menu of Printers
    D.  Custom installation of Printers

    E.  Menu of WordStar Features
    F.  Operating System considerations

    X.  Exit from Install

    Enter the letter of your choice.
```

Note:

At any time during installation, you can press ^C to exit immediately.

Selecting Printers Menu

Enter C to see the selection of printers available for your choice.

You will be shown one of two menus listing a dozen printer types. Some are specific brand names, some generic printer types. You may need the assistance of an expert on printers, your dealer, or the printer documentation to discover what kind of printer you have. Review also the Appendix 2, Connecting a Printer, in this volume.

If your printer is listed on this menu, press the appropriate letter and follow the instructions to return to the Installation Menu. If it is not, enter the number 2. The second printer menu will be presented with another list of printer types.

If your printer is on this list, enter the appropriate letter and follow the instructions to return to the Installation Menu. The INSTALL program will install that printer type's requirements into WordStar.

If your printer is on neither list and does not operate similarly to any printer on either list, press RETURN several times until you are returned to the Installation Menu. Then press D to select a custom installation for your printer.

Custom installation of a printer is not a Basic Level operation. Get the advice of your Osborne dealer. The custom installation procedure which follows the D selection in the menu is prompted. Required codes will be in the printer's manual. You will need to enter values in ASCII, in hexadecimal, or in decimal notation for such codes as the printer's Initialization code, Overprinting code, Backspacing code, and so on.

Protocol

As you exit the Printer Selection process, you will be asked to enter a protocol for the printer's communications.

Your WordStar is set up for no protocol since most Centronics devices use protocols built into the cabling system. Your printer may require one of the other two protocols, ETX/ACK or X-ON/X-OFF; see your printer's manual for this information. If no protocol is mentioned, press RETURN to skip this list.

Printer Driver

You will then be asked to select the "printer driver" program. Your WordStar is already set up for Primary List Device (selection A). This would only be changed under rare circumstances involving specialized use of your Osborne Executive. If you are not customizing your Executive, skip this list by pressing RETURN.

If your printer was on one of the prints lists, it should now be properly installed. Simply press X to exit the installation process. You will be asked to confirm that the newly re-installed WordStar be saved. Select "save changes." Press RESET, put WordStar in drive **A**, and press RETURN.

Test your printer by printing out the test program PRINT.TST. If the printer does not operate properly, check with your dealer regarding the cables and the switch settings in the printer. See also *Appendix 2, Connecting a Printer* regarding cables and printer switches.

Quitting

When you end your installation session, you will be given three choices:

1. You may SAVE the changes made during the installation; those changes will take effect when you press RESET and start WordStar again from the Sign-On message.
2. You can CHANGE the installation choices you made; this selection will return you to the Installation Menu.
3. Or you can QUIT and not save any changes. This is the selection that allows you to view the installation procedure and practice or explore the choices, without worrying about accidentally missetting one of the switches.

Install

Intermediate Level

Changing WordStar Defaults
WordStar Function Keys

WordStar Features

The default values for most WordStar features can easily be changed by using the INSTALL program.

Such features as insert mode, right justification, hyphen help, right and left margin, and top and bottom margin defaults can all be changed to new defaults you select to meet your needs.

It is advised that you spend some time familiarizing yourself with WordStar so that you will have a good idea of your own requirements before altering the defaults. Of course, defaults can be altered again simply by running the INSTALL program later. You can also make extra WordStar disks, each designed with defaults suited to a particular application.

To alter the WordStar defaults, first start the INSTALL program following the procedure listed in the Basic Level.

When you reach the Installation Menu, select:

```
        E.  Menu of WordStar Features
```

You will then be offered a list of the following WordStar features, each of which you may alter. Select the feature desired by pressing the appropriate letter. The alternatives are described on-screen.

Note:

You are changing the DEFAULT value of the feature named, which is the value WordStar uses when it starts an editing session. The feature can always be changed during editing, in the usual way. For example, the default for the Help Menu is level 3, which you can change while editing. If you change the default to level 0, that is the level at which WordStar will start an editing session; but you can always turn on level 3 in the usual way by entering ^ J H 3 *to view the help menu in detail.*

A ***Help Level***

Default: 3. WordStar initially presents you with the third and most explicit Help level each time you start. You may select a lower level as you become more familiar with WordStar. The choices are 0, 1, 2 and 3.

B ***Decimal Point***

Default: Period (.). In Europe the character used as a decimal is a comma; the period is used to signify thousands (100.000 instead of 100,000). This item allows you to enter another character; decimal tab mode will follow this character.

[C] *Non-Document Mode*

Default: Document Mode. If you exit WordStar and then re-enter by typing "WS," you can specify a file name on the same line, thereby immediately going to that file. For example: WS B:Filename will send you directly to that file as WordStar loads. You are automatically in Document mode in this case. Change this default to edit in Non-Document mode instead.

[D] *File Directory*

Default: ON. In the Opening Menu, the file directory is usually on. This command allow you to turn it off.

[E] *Insert Mode*

Default: ON. When you first enter a file, Insert mode is usually ON. To have it usually OFF, use this feature.

[F] *Right Justify*

Default: ON. WordStar usually right-justifies text unless you turn Justify OFF. Use this feature to change the default to ragged right.

[G] *Hyphen Help*

Default ON. May be changed to OFF.

[H] *Page Number*

By default, WordStar numbers every page unless instructed to omit them (with the .OP command). Change this default to make WordStar automatically OMIT page numbers unless told to use them (with the .PN command).

[I] *Top Margin*

Default setting for the top margin (how many RETURN WordStar comes down from the top of the page before beginning to print) is 3. Enter another number to change this default setting. (The .MT command will still override this setting.)

J Bottom Margin

Default setting for the bottom margin (how many RETURNS short of the bottom WordStar ends a page) is 8. Enter another number to change the default. (The .MB command will override.)

K Left Margin

The left margin setting is normally 1 (begin in column 1). Change the default left margin setting by entering another number.

L Right Margin

The default is a right margin of 65. Enter another number to change this default.

M Lines Per Page

WordStar defaults to 66 lines per page (6 lines per inch for 11 inch paper). Select another number to change this default if you regularly use different length paper, or if you set your lines-per-inch at a value other than 6.

N Page Offset

This is how many characters in from the left side of the paper the left margin column 1 begins. Default value is 8 characters. (Note: this is the value that determines where the LM setting of 1 begins.)

O Form Feeds

The Print Options question "Use Form Feeds?" is normally answered NO if you skip the print options. Turn Form Feeds ON and your default value will automatically cause WordStar and MailMerge to pause between pages to allow you to change paper during printout.

[P] *Data Field Delimiters*

MailMerge looks for a comma in the data file to determine where one variable ends and the next begins. This feature allows you to change that delimiter to another character (perhaps to conform with another program that produces data files in a different way).

[Q] *Keyword Symbol*

MailMerge knows a keyword in a form letter because it is bracketed by the ampersand: &. This feature allows you to change that keyword symbol to another character if you need the ampersand for something else.

[R] *System Disk Drive*

Allows you to keep your .OVR overlay files on other than the **A** drive disk (in other words, on a disk other than your WordStar disk). WordStar normally checks the logged drive, then the **A** drive to find them. It is not normally recommended that this be done.

When you have finished customizing your WordStar features, press [X] to return to the Installation Menu and then SAVE your newly changed defaults to the WordStar disk. You must press RESET and start WordStar from the Sign-On message in order to make the new values begin to work.

Function Keys

When you have become familiar with WordStar, review the explanation of Function Keys given in the SETUP section of *Volume 1—Mastering the Osborne Executive.*

The ten key pad on the right of the keyboard can serve as a group of function keys, also called "soft" keys or F keys on some computers.

The SETUP program in CP/M Plus can set up these keys so that, when combined with the CTRL key, they will issue common WordStar commands.

Experience with WordStar will tell you which commands you use most often, and which would most benefit from being set up in a function key. Here are some common WordStar commands that experienced users most often set up in their function keys:

Command	**Function Key**
Log the B drive:	L B RETURN
Paragraph Indent:	^OL10 RETURN ^OR 55 RETURN
Check Disk Space:	R DIR *
Save & Quick Return:	^KS ^QP
Quick Reformat:	^QQ ^B 1
Page Turner:	^QF ^N RETURN 55 RETURN
Merge-Print Print File:	M PRINT.FIL ESC RETURN

Also consult **The Portable Companion** magazine which contains the distilled experience of Osborne users the world over and is an ideal source of more information on efficient use of the Osborne Executive and its programs.

Install

Advanced Level

Terminal Menu
Operating System Menu

Terminal Menu

The Osborne Executive is set up to emulate the Televideo 912 and is already installed.

If the computer is later changed to emulate another terminal type, the Terminal Menu provides a selection of some 28 standard terminals to allow WordStar to operate easily in these modes.

The Custom Installation menu allows any terminal type not listed in the A menu to be installed for WordStar by entering the values called for.

Both of these menus are completely menu-driven and require only the entry of the specified ASCII, hexadecimal or decimal value as indicated on screen.

Printer Install

If a printer is installed that uses other than the Primary list device and other than the parallel port on the Osborne Executive, you must use the SETUP program found on the CP/M Plus System Disk to adjust the operating system's print routines accordingly. The SETUP program is described in *Volume 1 — Mastering Your Osborne Executive* and is also menu-driven.

Operating System

The Operating System Considerations menu simply asks if the computer is being used in a multi-user MP/M environment instead of the CP/M Plus environment which is usual for the Osborne Executive.

APPENDIX 2
Connecting a Printer

Introduction

This Appendix contains general information which can help you perform your own printer installation.

Important Note:

It is NOT necessary for you to be able to perform your own interface. If you do wish to perform the interface yourself, this guide will give you the information you require. It is not intended, however, as a complete and detailed tutorial on electronics.
If you are new to computers, you are advised to take your printer and your Osborne Exective to your Osborne dealer, and he will perform the interface for you for a fee.

There are hundreds of printers on the market, and each one seems to have its own unique way of interfacing with each computer. At Osborne, we have performed interfaces on dozens of printers to gather information for this guide.

Type of Interface

The first step in performing the interface is to decide which kind of interface you wish your printer to use.

There are two main types of printer interface: parallel and serial. We will explain the difference between them in a moment.

Since the Osborne Executive has ports for connecting either type of interface, first determine what interface the *printer* requires.

Some printers allow you to choose either interface. From the standpoint of performance, reliability and flexibility, there is little reason to prefer one over the other. Instead, your choice will depend on other factors, such as:

- Will you want your printer more than eight feet away from your computer? Then choose the serial connection, because a parallel cable should not be more than eight feet long, while the serial cable can be much longer.
- Do you plan to get a hard disk some day? The hard disk will plug into the parallel port (IEEE-488), so plug the printer into the serial (RS232) port.
- Will you need to plug in scientific test instruments to your computer? These use the IEEE-488 port also, so your printer will go to the serial port.
- Are you planning to do the installation yourself? Many people find the parallel interface cabling easier to work with.

There may be other considerations as well, so discuss your decision with your printer vendor.

Now we will explain some of the theory of both parallel and serial interfacing, and the information you will need to perform the interface successfully.

First, we will explain how to interface your serial printer with the Osborne Executive RS232 port.

Note:

Even if you are getting a parallel interface, read this next section anyway, since many of the instructions are similar for both kinds of printers.

RS232 Serial Printer Interface

The Electronic Industries Association (EIA) developed an interface standard around which most business equipment manufacturers could design the connections for their computers, printers and other products. This widely-distributed standard is called the EIA RS232C standard.

The RS232 port sends out electrical signals one at a time in a series, one after another, like a train. Thus the interface is referred to as SERIAL (as opposed to the PARALLEL interface, whose signals go out in parallel groups, along a parallel wire).

The serial port sends data over a line one bit at a time. The receiving device collects the incoming data bits and assembles them into usable groups.

With the Osborne Executive, the serial data transfer group is 8 bits long. A bit is one signal (or "signal event") and one 8-bit group defines one character (a letter of the alphabet, a number, or a one-instruction code such as "insert carriage return").

The 8-bit group is packaged with a start bit at the beginning and a stop bit at the end. The start bit signals that a new package is coming. The printer then collects the next 8 data bits and assembles them into a code standing for a letter or a instruction (by checking against a chart it stores internally).

To check that the transmission was received correctly, the printer then waits for the last stop bit. If that arrives, the printer translates the group and prints out the appropriate letter on the page (or performs the instruction ordered, such as underlining or scrolling).

Some computers (but not the Osborne Executive) send yet another bit, called the parity bit, which is used to double-check the transmission.

Because each bit group is monitored individually with the start and stop bits, there can be a certain amount of irregularity in the timing of the individual bits without affecting the performance. For this reason, this kind of transmission is called "asynchronous."

The BAUD rate is the number of bits per second sent down the wire. You will almost always use a baud rate of 1200. Most printers will accept this baud rate or even higher baud rates. Some much older printers or specialized output devices (such as telephone modems) can only accept 300 baud or (in the case of telex transmissions, for example) 110 or even 50 baud. For the most part, however, you will use 1200 baud with your printers.

The reason you need the preceding information is to set your DIP switches on your PRINTER. A DIP switch is a miniature switch (see your printer manual for their location) which you can use to choose a certain interface.

Protocols

You will also have to discover and interface with your printer's protocols. A protocol is "a set of rules governing the exchange of information between two systems" — in this case, they are the signal codes that the printer sends.

What signals would a printer send to a computer?

First, whether the printer is at a ready status (turned on and ready to accept information for printing (also known as SELECTED).

Second, to tell the computer to stop sending signals until the printer catches up. Remember, the computer sends the printer signals at 1200 baud, or 120 characters of type per second. The printer, on the other hand, prints at 12 to 200 characters per second. The printer lags behind, storing the incoming data in its memory buffer. When the buffer fills up, the printer signals the computer to stop sending until the buffer can be emptied. The larger the printer's buffer, the less often the printer needs to stop the computer.

The third signal a printer would send is to tell the computer to start sending again.

And finally, the printer will send a stop signal to the computer if it runs out of paper, or when the printer cover is off, or when it reaches the bottom of the page while in "stop after page to insert paper" mode; and (in some printers) when the ribbon runs out.

Printers have various ways of signaling the computer. One is called the "hardware handshake." A handshake is an acknowledgement. With this system, the cable itself utilizes an RS232 signal called "reverse channel" or "secondary request to send," whose details we won't go into here. Hardware handshaking is often preferred when multiple protocols are available because it greatly simplifies the interfacing procedure.

The next one is the protocol called XON/XOFF. When the printer needs to stop the computer, the printer sends a special signal called X/OFF, also called DC3 control code. The Osborne Executive then stops transmission of data. When ready, the printer sends an X/ON control code and the Osborne Executive picks up where it left off. XON/XOFF has a wide variety of Install options and is able to choose many printer special features.

Finally there is the ETX/ACK protocol system. With this common system, the Osborne Executive marks the end of each line it sends with a special End of Text (ETX) signal. When the printer withdraws this line from the buffer, it sends an Acknowledge (ACK) signal back to the computer, indicating that the computer can send another line of data.

Now that we understand the theory, let's perform the interface of our serial printer. There are four steps.

Step 1: The Printer

First, we must set the DIP switches. (Note that some printers use JUMPERS instead of DIP switches). You may wish to check with your Osborne dealer to see if your printer, and its DIP switch settings, is listed in his Osborne Application Notes.

Since the Osborne Executive uses a start bit, eight data bits, a stop bit, and no parity bit with 1200 baud rate, that's what you will set the DIP switches for.

Look in your printer's manual for the location of the DIP switches. They will be shown on the machine and each switch identified. Not all the switches listed here will actually be present on the machine. Look for the following switches and set as indicated:

a. START BIT: A few computers offer a choice of one or two start bits. Choose 1.

b. DATA BITS: You have a choice of seven or eight data bits. Choose 8.

c. STOP BIT: Again, some printers let you pick one or two. Pick 1.

d. PARITY: Your choices will usually be Even Parity, Odd Parity, and No Parity. Choose NO Parity.

e. BAUD RATE: Your printer may offer a variety. Choose 1200 where possible. Some printers, like the Smith-Corona TP-1, are preselected at the factory for higher baud rates (9600 in the case of the TP-1), in which case you can leave it at the higher rate.

f. MODE: Some printers offer a choice of SERIAL or PARALLEL. Select SERIAL (if you are using the RS232 Port and a Serial Printer option), or PARALLEL (if you are using the IEEE port and the Centronics/Parallel option described later in this Appendix).

Now for the Communications Protocol. Choose only ONE of the following:

g. XON/XOFF: If you want to use this protocol, set it to ENABLE. If not, then set it to DISABLE.

h. ETX/ACK: Again, if you want to use this protocol, set at ENABLE; otherwise, DISABLE.

Some printers may offer something called "RDY/BSY," which is a Hardware Handshake protocol (Ready/Busy). Choose that one if it is offered, instead of XON/XOFF or ETX/ACK. It will probably have been preset to ENABLE at the factory anyway.

i. LINE FEED: Automatically inserts line feed when carriage return signal is received. Set this switch OFF.

j. DEVICE ADDRESS: Set OFF or INVALIDATE.

The line feed switch is an important one. If you don't have it off, you will end up with doublespaces after every line you print. Printers offer an automatic version because some computers don't send a "carriage return" character and a "line feed" character both, but just a "carriage return" character. The printer will automatically assume the line feed every time it gets the carriage return.

The Osborne Executive, however, always sends both characters, so the automatic line feed option is not needed except in special applications. Failure to turn this switch (if present) to OFF will result in an inability to use the BOLDFACE print function.

There may be other DIP switches or Jumpers on your printer, but these are the most important ones. You can discover the uses of the others by experimenting after these have been set and tested.

Step 2: The Cable

If you know something about making your own cables, the following diagrams should be of help to you. Alternately, your dealer will be able to wire the cable you need from this information.

The connections are as follows:

Executive		Serial Printer	
RS232 (DCE)		(DTE)	
(Data Comm.		(Data Terminal	
Equipt.)		Equipt.)	PURPOSE
GND 1	- - -	1 GND	Frame ground (optional)
TXD 2	- - -	2 TXD	Transmitted Data
RXD 3	- - -	3 RXD	Received Data
RTS 4	- - -	4 RTS	Request to Send
CTS 5	- - -	5 CTS	Clear to Send
DSR 6	- - -	6 DSR	Data Set Ready
GND 7	- - -	7 GND	Signal Ground
DCD 8	- - -	8 DCD	Received Line Signal Detected
DTR 20	- - -	20 DTR	Data Terminal Ready

If your printer is able to use reverse channel or one of the other hardware handshake signals, then the cable will look the same as above EXCEPT for the last line, #20, Data Terminal Ready, which will look like this instead:

DTR 20 - - - 11 SCA Secondary Request to Send
OR: 19 RTS Request to Send

This is a hardware handshaking cable. Pin 11 is an output from the printer used as a terminal ready/busy indication for computer peripheral applications, and is also known as "reverse channel." Pin 19 is another hardware handshake line on certain printers.

Note that all signal names in this standard are given from the point of view of the DTE device (the printer). Received Data (line 3), for example, refers to the *printer* receiving data. That is why pin 3 is called Received Data both on the computer and on the printer, why you connect Received Data to Received Data and Transmit Data to Transmit Data, and why you don't connect Transmit Data to Received Data.

Step 3: Osborne's Setup Program

Now you need to tell the software what printer configuration you are using, by running your CP/M Plus SETUP program. Details on using this program are *in Volume 1 — Mastering Your Osborne Executive.* The following sections of SETUP will be affected:

Printer Options

Go to the Printer Port selection in SETUP.

Your choices are:

- No Protocol (Standard)
- ETX-ACK (for Qume or NEC printers)
- XON/XOFF (for Diablo printer types)

Choose the one corresponding to your printer type.

If your printer is using a HARDWARE HANDSHAKE, or a BUILT-IN HANDSHAKE (a form of hardware handshake), or if you are running at 300 BAUD, then choose No Protocol.

(If you run your printer at 300 baud, the buffer will not usually overflow; therefore, handshaking and acknowledging are not really necessary.)

Other options listed are for parallel printers, which we will get to in the next section.

Baud Rate

Set your baud rate for either 300 or 1200, depending on the capability of your printer. Again, as a rule you will prefer to use at least 1200 baud if your printer allows it.

Then select PRINTER(LST:) as the device to be attached to the port.

Follow the remaining instructions in the SETUP section of *Mastering Your Osborne Executive* carefully to save the Setup to the appropriate disks. Don't forget that you must save the new Setup and then reload the program before the new Setup values are activated.

Also note that you will have to change the SETUP program for each disk on which you have used SYSGEN to transfer the CP/M Plus system, so that each program disk you use (especially your WordStar disk) will have the same Setup parameters. If you later discover that the printer runs on some disks but not on others, it could be that you forgot to copy the new Setup to some of your disks.

Testing the Printer

We will now make a preliminary test to check that the printer will work with the cable and DIP switches and Setup changes we've made. We will do this before using WordStar's INSTALL program (Step 4) to avoid certain complications which WordStar introduces.

The easiest way to see if the Osborne Executive can talk to the printer is to use the Control P (^P) function in CP/M Plus.

Hook up your cable, turn the printer on, check the paper supply, and check the printer lights to make sure the ON-LINE or SELECTED light is lit.

Load CP/M Plus into the computer. When you have the CP/M Plus prompt A>, type CTRL P D I R RETURN (which stands for Print a listing of the files on the disk).

If everything is all right, the printer will start printing a listing of your disk directory.

Now enter another ^P to toggle the Print function off.

Printing from the CP/M Plus level, which you just did, is the simplest type of printing for the computer and should be used whenever you want to establish that there is a functioning communications pathway between the printer and the computer.

If the printer DOES NOT print; or if it prints, but does odd things like print random characters, print in strange places, or otherwise do the unexpected, you will have to check your cable, your setup program, your DIP switches, and review these instructions. Also check your Osborne dealer's Application Notes. If your printer is listed, more details on its performance may lead you to the problem.

If you get desperate, you can experiment by flipping DIP switches or trying other SETUP commands and see what happens. Or you can just take your machinery in to your Osborne dealer to locate the problem.

Step 4: WordStar Install Program

INSTALL is only used for WordStar, so you don't have to worry about this step for the other software.

Check *Appendix 1 — Modifying WordStar With Install* for details on how the Install procedure works change INSTALL to reflect the same printer choices used in SETUP.

Important Note:

There are very few printers that REQUIRE additional modifications to the WordStar Install program. Much of it is purely optional.
WE STRONGLY RECOMMEND THAT YOU DO NOT CHANGE WORDSTAR'S DEFAULT SETTINGS in Install until you have at least observed how the printer operates with those default settings.

This is simply good interfacing technique: do one step at a time, check the printer output, then make another change if necessary. To change your WordStar settings before seeing how the printer works for the first time is like adjusting the timing on an unfamiliar car without even knowing if it runs first.

Should you need to customize the default settings, here is a guide to some aspects.

Line Feed

Printers will automatically insert a line feed when they receive a carriage return instruction from the computer.

Most printers have a DIP switch that allows you to turn off this automatic feature, which we have advised you to do earlier because the Osborne Executive always sends a line feed instruction when one is needed. If you did not disable the automatic line feed function, you would automatically get double-spaced copy whether you wanted it or not.

Some printers, however, do not have a method of disabling the automatic line feed. The Smith-Corona TP-1 series, IBM I/O Selectrics, and certain Radio Shack models are examples. Check to see if your printer does have a switch for this function, and turn it off if you can.

The automatic line feed function creates problems when you try to BOLDFACE your material. Boldface is performed on most printers by printing out the line, then carriage returning WITHOUT a line feed, and printing on top of the same line a second and a third time to make it darker. The result looks like this:

A BOLDFACE line is struck three times.

A printer with an automatic line feed, however, ends up printing each line three times instead, since each carriage return automatically introduces a line feed. Thus such a printer would print the line like this:

> A BOLDFACE line is struck three times.
> A BOLDFACE line is struck three times.
> A BOLDFACE line is struck three times.

It will also permit you to get single-spaced printing out of your printer.

To Install such a printer (one with a locked-in automatic line feed), pick this item in the Printer menu:

> "BACKSPACING STANDARD PRINTER"

With this setting, the printer will type each character, then backspace once, type the same character again, backspace again, and type the same character a third time. That way you will still be able to get your boldface ability.

If you CAN disable the automatic line feed on your printer, leave the default setting at "ANY STANDARD PRINTER."

Special Functions

A true SUPERSCRIPT prints slightly above the line.

A true SUBSCRIPT prints slightly below the line.

Both can be incremented to various degrees.

In MICROJUSTIFICATION, all characters are allocated space on the page in accordance with their size. Thus the letter "M" takes up more space than the letter "I", the letter "K" more room than a period, a small "e" less than a capital "E." The result is printing that looks like it was typeset.

The bad news is that, as of this writing, WordStar DOES NOT SUPPORT PROPORTIONAL SPACING in the versions 2.26 and 3.3.

Accessing other functions will require experimentation on your part. Some printers cannot do incremental rolling. Some cannot reverse tractor feed, which is required for true superscripting and subscripting. If you Install your daisywheel printer using the DIABLO 1640/1650 option or the QUME SPRINT 5 option, you might achieve scripting and microjustification. Experiment and try for yourself.

IEEE-488 Parallel Printer Interface

You will recall that serial printers receive data bits one at a time on a single line. In parallel transmission, each of the eight data bits constituting a byte (or character) of information has its own wire to travel on, and each group of eight travels down the flat connector cable in parallel, like automobiles on an eight-lane highway.

There are other data bits also transmitted along with the eight-bit character group. In addition to the eight data bits, the typical parallel interface (generally called the ''Centronics-type interface''), has four more wire types:

1. A Data Strobe
2. A Busy Signal
3. A Select
4. A number of Signal Grounds

The printer sends a Printer Ready (or Selected) signal down line number 3. The Osborne Executive then sends out a character using the eight data lines.

To allow for the slight irregularities in travel time of the eight bits, the Osborne Executive then sends out a Strobe signal, which the printer uses to mark the end of a character transmission. The printer then checks the data bits to make sure they are all present and accounted for, translates the signal into a character, and prints the character.

When the printer's buffer is full or the printer is ready to print out a single line, the printer sends a signal down the Busy line telling the Osborne Executive to wait until the printer is ready again.

The Signal Ground wires are used to protect certain of the wires from random electronic interference.

Now let us consider the steps in setting up your parallel printer. First, however, be sure to read the Serial Printer section.

Step 1: The Printer

One nice thing about the parallel interface is that it requires fewer DIP switch settings.

If your printer has a choice between serial and parallel interfacing, check the printer manual for directions of setting the printer up for parallel interfacing (in this case). Usually this involves setting a DIP switch or a jumper as shown in the Serial section.

Next, set the "Automatic Line Feed Upon Carriage Return" DIP switch, if there is one, to OFF.

And finally, if there is a DIP switch corresponding to "Device Address," turn it to OFF or INVALIDATE.

And that should be it. The parallel interface does not use start, stop and parity bits.

Step 2: The Cable

We recommend that you do not make your parallel cable more than eight feet long.

The reason is that although the eight bits of the character group are sent out at the same time from the computer, they don't all arrive at the printer at the same time. Slight irregularities in the wire diameters and metal composition, interference from neighboring wires and stray electrical fields, cause the eight bits to arrive almost, but not quite, simultaneously.

The Strobe bit is sent as a pacing device, to enable the printer to certify that the group has all arrived. But as the cable gets longer, the discrepencies in the arrival times of the bits (called the "skew rate") reaches the point where the printer can no longer cope. Eight feet is about as long as the cable can go and still perform reliably.

Plugs

Parallel printers are called Centronics-compatible printers, after the brand name of a popular parallel printer for microcomputers.

The parallel cable consists of a flat, ribbon-like cable with connectors, or plugs, at both ends. The first thing you need to do is find out what kind of plugs to put at the ends.

The Osborne Executive's port for interfacing with the parallel cable consists of a 25-pin GPIB plug. So one end of your cable will have this plug on it to hook up with the Osborne Executive IEEE-488 port.

Now for the other end of the cable.

Virtually all Centronics-compatible printers we have tested use a Centronics Connector, which is similar to the GPIB plug. These are also standard plugs.

There is one printer we have found, however, that uses a card-edge connector at the printer. In that (rather rare) instance, you would have to get a cable with a card-edge connector at one and our 25-pin GPIB plug at the other.

Here is a diagram of the layout of the wires on the flat ribbon parallel cable and their assigned functions. This cable layout works with the majority of "Centronics-compatible" printers.

Executive IEEE-488 Port Pin #	Function	Printer-Centronics Connector Pin #
1	DATA 1	2
2	DATA 5	6
3	DATA 2	3
4	DATA 6	7
5	DATA 3	4
6	DATA 7	8
7	DATA 4	5
8	DATA 8	9
10	GROUND	30
11	STROBE	1
12	GROUND	19
15	BUSY	11
16	GROUND	29
18	GROUND	21
19	SELECT	13
20	GROUND	20

Step 3: Osborne's Setup Program

See the Serial Interface discussion and the SETUP program section earlier in this Appendix.

The only thing you need to do in the Setup section is specify CENTRONICS under PARALLEL PORT. You don't need to bother with baud rate with a parallel printer. Save the Setup to the disks you will be using.

Now test your printer by running through the Print From CP/M Plus test discussed earlier in the Serial Interface section. Everything should now work just fine.

Step 4: WordStar Install Program

As mentioned earlier, you should not tinker with your WordStar default settings unless your printer gives you reason to.

Also as mentioned earlier, the most common printer problem that might cause you to change the Install default settings is the failure of the printer to print BOLDFACE, giving triple lines instead of one extra-dark line. In that case, change the default Item C from "STANDARD PRINTER" to "BACKSPACING STANDARD PRINTER."

If other problems arise, as mentioned in the Serial Interface section you will have to experiment with the Install settings in order to overcome the defect in the printer operation.

Application Notes

There are dozens of printers on the market and many of them have been manufactured with original and unique methods of interfacing with the dozens of microcomputers on the market.

Unfortunately, the printer documentation does not always make it clear how to perform this interface.

In our effort to assist you, we have performed and continue to perform interfaces on a variety of popular printers. The resulting Application Notes are sent to all Osborne dealers regularly. A list of some printers for which Application Notes are available is below. See your Osborne dealer for more information.

Index of Printers:

Smith Corona TP-1 parallel
Smith Corona TP-1 serial
NEC 8023A-C parallel
C. Itoh 8510A parallel
C. Itoh F10-40 serial
Transtar serial
C. Itoh Starwriter 25/45 serial
Diablo 630 serial
Okidata Microline 82-A parallel
Epson MX-80 parallel
NEC 3500 and 7700 serial
NEC 7700 parallel
NEC 7730 parallel
Diablo 1610/1620 serial
TI Omni 80 Series serial
Centronics Cabling Theory

PART 3
SuperCalc

Introduction

SuperCalc lets you work with numbers in much the same way WordStar lets you work with words. This material assumes you have an introductory-level understanding of the Osborne Executive Computer and CP/M.

What is SuperCalc?

SuperCalc performs common business calculations, such as planning, forcasting, and developing investment strategies. You can use SuperCalc to explore possibilities and compare alternatives in your decision-making process.

The program is simple, straightforward and easy to use. SuperCalc presents you with a worksheet where you can enter data. SuperCalc performs the computations for you. If you change any of the information, SuperCalc automatically recomputes the data.

SuperCalc stores the data on the worksheet. You can display, edit, and print these worksheets at a later time.

Using this Manual

Here are some suggestions on how to use this manual. Each cluster of functions, such as Moving around SuperCalc, has three divisions: Basic Level, Intermediate Level and Advanced Level. Select the appropriate level for your background, and refer to the material covered in that level as you learn about SuperCalc.

If you are using SuperCalc for the first time, you should follow the instructions on SuperCalc in the Beginner's Guide (Volume

0) first. In the rest of this manual, read the topics covered under the Basic Level. As you gain more familiarity with SuperCalc, you may want to refer to the Intermediate and Advanced levels for additional information.

If you are already familiar with SuperCalc, you may want to scan through the Basic Level discussions, and then refer to the discussion in the Intermediate level when you start creating your own worksheets.

After you develop expertise with the concepts discussed at the Basic and Intermediate Levels, you may wish to find out about the more sophisticated options and techniques discussed under the Advanced Level.

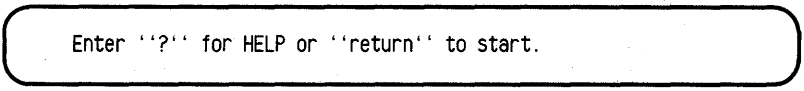
In summary, this manual is designed to:

> allow the beginner to perform common business calculations with SuperCalc.
>
> allow the intermeditae level user to perform more complex calculations.
>
> guide the advanced SuperCalc user toward a sophisticated level of familiarity with this powerful electronic worksheet.

Starting SuperCalc

1. Press the RESET button on the front of your Osborne Executive.
2. Place the SuperCalc diskette in drive A, close the drive door, and press RETURN.
3. The Osborne logo will appear on the screen. After a few moments, you'll see the SuperCalc sign-on message.

At the bottom of the screen is a line that reads:



```
Enter ''?'' for HELP or ''return'' to start.
```

Press RETURN.

The SuperCalc worksheet appears on your screen.

Pressing **?** from the sign-on screen will give you some general information about SuperCalc. You might want to try this at some other time. For now, you should examine the worksheet on your screen and read the information in the next section, "General Concepts"

SuperCalc can also be started from the A prompt by typing **SC** and pressing RETURN.

General Concepts

Basic Level

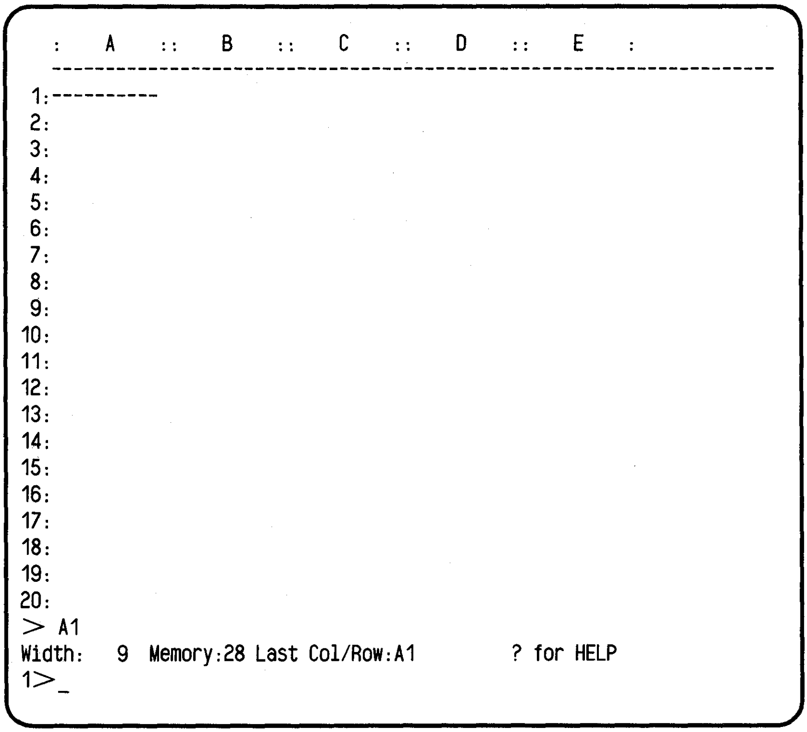
The Worksheet

The SuperCalc worksheet is divided into rows and columns. You see columns A through H indicated on the first line of your screen. The rows 1 through 20 are indicated on the left side of your screen.

The entire SuperCalc worksheet has 63 columns and 254 rows. The rows are numbered 1 through 254. The columns are lettered A-Z for the first 26, AA-AZ for the next 26 columns, and BA-BZ for the last 26 columns. What you see on your screen now is a "window" showing you only part of the entire worksheet. You can view any portion of the worksheet by moving your cursor to it. We will show you how to do this in "Moving Around SuperCalc."

A Cell

The intersection of a row and a column in the worksheet is called a "cell". Every cell has a unique letter-and-number combination assigned to it. When you start SuperCalc, your cursor is in Column A, row 1. This cell is called A1. The cell in the lower right hand corner of your screen is H20.



```
  :   A   ::   B   ::   C   ::   D   ::   E   :
  --------------------------------------------------------------
 1:---------
 2:
 3:
 4:
 5:
 6:
 7:
 8:
 9:
10:
11:
12:
13:
14:
15:
16:
17:
18:
19:
20:
> A1
Width:   9  Memory:28 Last Col/Row:A1        ? for HELP
1>_
```

The Active Cell

You enter data into the cells of the worksheet. You can only enter data into one cell at a time. This cell is called the active cell. When you type in data, it goes into the active cell. The row and column that contain the active cell are called the current row and current column.

When you first start SuperCalc, the worksheet is empty. As you use the worksheet, your changes are shown on the worksheet and stored in the memory of your Executive. These changes include putting information into a cell, formatting a cell, or using a cell in an equation. Cells between two cells you have used are also stored in the Executive's memory. Cells to the right or below the area you have used are not stored in the memory. This allows you to choose how much of the worksheet you want stored.

Two Cursors

There are two cursors on your screen. The long, bright cell within the worksheet is called the worksheet cursor. It shows you the active cell.

The smaller cursor on the last line of your screen (next to the 1>) displays the characters you type before they are entered into your worksheet.

Status Line

There are three lines at the bottom of the SuperCalc worksheet. The top line is called the status line. It tells you the active cell. With your worksheet cursor in the upper left corner of the screen, the status line reads >**A1**.

The first character in the status line tells you the direction you are moving through the worksheet. Your direction changes each time you press an arrow key and is reflected by the "pointer" at the start of the status line. This pointer tells you whether you are moving up (^), down (v), left (<), or right (>).

If the active cell contains information, this will also be displayed on the status line. This information can be numbers, text, or formulas.

For example, if the number 42 were in the active cell A1, your status line would look like this

```
>A1     Form=42
```

The word "Form" tells you that the information in the active cell is a formula or a number. If the cell contained text, this would be reflected by the word "Text" in the status line.

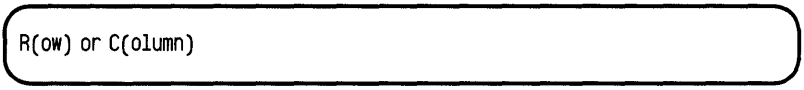
The Command Line/Global Status Line

The command line is the second line below the worksheet. When you first start SuperCalc, it will look like this.

Width: 9 Memory: 34 Last Col/Row:A1 ? for HELP

This line displays the current cell width, the amount of memory available to use, and the last cell you used. It also reminds you that you can press **?** for some general information.

The command line is also used to display prompts. When you are entering SuperCalc commands, this line displays a list of options that are available to you at that point. For example, after you have issued the **Delete** command, this line shows the Delete prompt.



```
R(ow) or C(olumn)
```

Command-Entry Line

The last line on your screen is where you enter and edit information before you press RETURN to enter it into your worksheet.

General Concepts

Intermediate Level

Status Line *Command-Entry Line*
Command Line/Global Status Line

Status Line

Here is an example of a more complex status line:

> A5 L$TR P Text = "February

Here is what it means:

> is the current direction of the worksheet cursor, set by the last arrow key pressed. It may be >,<,v, or ^.

A5 is the active-cell location. Data entered will go into that cell. Commands that use the current column or row will use the column or row containing that cell; in the example column A and row 5.

L$TR shows the active-cell format settings; numbers are left-justified, $ is the format, and text is right- justified. (The detailed description of the /Format command gives full information on these settings).

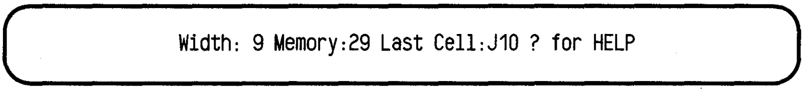
P shows data protection of the active cell. This area is blank if the cell in unprotected.

Text="February indicates the contents of the active cell, in this case, text. "Rtxt=' indicates repeating text. Numbers or formulas are shown as **"Form"**; for example, "FORM=12*B9."

SuperCalc also uses the status line to display error messages and certain information messages. These special messages disappear and the status information reappears when you press any key.

Command Line/Global Status Line

The second line below your worksheet shows you the global status. For example:



```
Width: 9 Memory:29 Last Cell:J10 ? for HELP
```

This information is:

Width:9 is column width. This is the display width of the column that contains the active cell. The standard, or default, setting is 9, but you can specify a different width. You can set all columns to the same width or set different widths for different columns. If you change the default setting, the status line lists the display width that you select.

Memory:29 is available memory in kilobytes (a kilobyte is the memory sufficient to hold 1024 characters or digits). This number changes as you add data to the worksheet.

Cell:J10 tells you that the area you have used on the worksheet extends from cell A1 to cell J10. In other words, J is the right-most column that you have used, and 10 is the lowest row.

? for HELP reminds you that pressing **?** will give you an explanation of the options you have at that moment.

As you enter commands, the command line will change to show you your current choices.

Command-Entry Line

The last line at the bottom of the screen is the command-entry line. The information you type appears on this command-entry line. The command-entry line is much like a scratch pad, in that it allows you to check and edit the data and text you wish to enter before you commit it to the worksheet.

As you type, the command-line cursor moves to indicate where the next character will appear. As your cursor moves, the number in the left-hand margin changes to reflect your cursor position in this line.

Moving Around SuperCalc

Basic Level

This section describes moving around in a worksheet.

Sample Program

To practice moving around in a SuperCalc file, we have provided a sample file called BRKEVN on your SuperCalc diskette. Follow the steps below to load this file. Loading a file is described in detail in a later section.

Turn your computer ON, or press the RESET button. Start SuperCalc by pressing RETURN. When the SuperCalc sign-on screen appears, press RETURN. A blank worksheet appears on your screen. To load the BRKEVN file, type the instructions below. The two last lines on your screen change as you type these instructions. These prompts are discussed in other sections of this manual.

Type / L B R K E V N

If you make a mistake, press the left-arrow key to backup and retype the entry correctly.

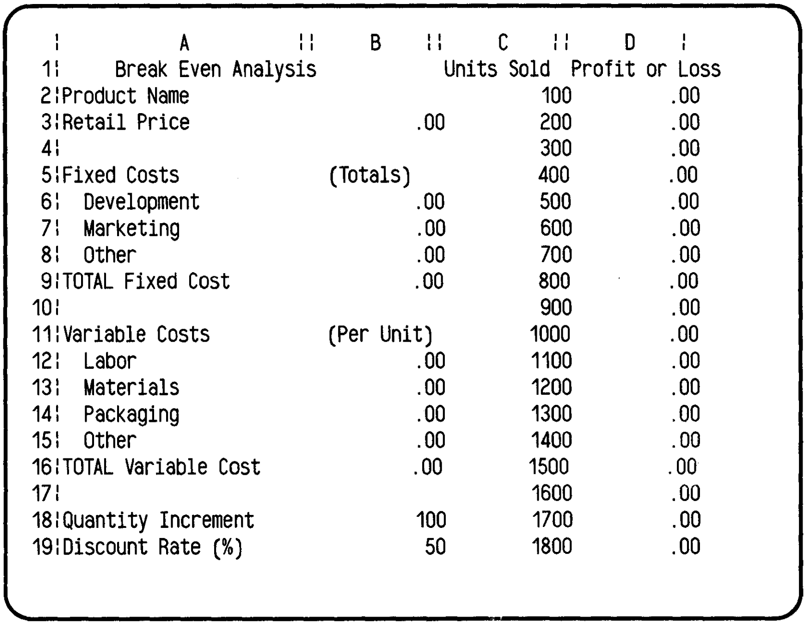
When it is typed correctly, press RETURN.

When the prompt

```
A(ll) or P(art)?
```

appears, press A to load all of the file.

In a little while, the BRKEVN worksheet appears on your screen.



```
  ¦          A          ¦¦    B    ¦¦     C     ¦¦     D     ¦
 1¦     Break Even Analysis          Units Sold  Profit or Loss
 2¦Product Name                             100            .00
 3¦Retail Price                  .00        200            .00
 4¦                                         300            .00
 5¦Fixed Costs           (Totals)           400            .00
 6¦  Development                 .00        500            .00
 7¦  Marketing                   .00        600            .00
 8¦  Other                       .00        700            .00
 9¦TOTAL Fixed Cost              .00        800            .00
10¦                                         900            .00
11¦Variable Costs        (Per Unit)        1000            .00
12¦  Labor                       .00       1100            .00
13¦  Materials                   .00       1200            .00
14¦  Packaging                   .00       1300            .00
15¦  Other                       .00       1400            .00
16¦TOTAL Variable Cost           .00       1500            .00
17¦                                        1600            .00
18¦Quantity Increment            100       1700            .00
19¦Discount Rate (%)              50       1800            .00
```

Arrow Keys

The simplest way to move the worksheet cursor is with the arrow keys: ⇧⇩⇦⇨.

Scrolling

When your cursor moves past the current window, those cells that are off-screen come into view, a column or row at a time. As you move the worksheet window either horizontally or vertically, you are "scrolling" the display.

Continue to "scroll" the screen with the left-arrow key until you pass column Z. Note that two-letter representations mark the remaining columns.

Setting the Direction

The arrow keys can also be used to set the cursor direction.

▽	Set the direction to Down
△	Set the direction to Up
◁	Set the direction to Left
▷	Set the direction to Right

After you press an arrow key and set the direction, pressing the RETURN key advances the worksheet cursor cell by cell in that direction. The status line indicates the direction the worksheet cursor will move when you press RETURN.

Move the cursor around the worksheet and watch the status line as the active cell and direction indicators change.

Going to a Cell

You can use the = command (also called the **GOTO** command) to move directly to a designated cell.

⊟ *Move to the Designated Cell*

Type ⊟. You are asked to "Enter the cell to jump to." Type the column and row of the cell to move to, and press RETURN. For example, type F 6 in response to this prompt and press

RETURN. The display on your screen changes. If the designated cell is already on the display, the worksheet cursor will move to it. If not, the window will move to show the new active cell at the upper-left corner of the screen.

If you type only = and press **RETURN**, the window will adjust to show the current active cell at the upper left of your screen.

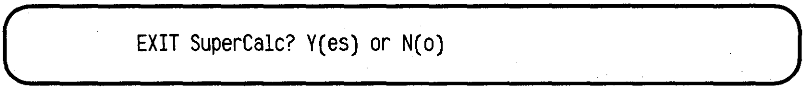
HELP Command

When you press **?**, SuperCalc explains some of the things you can do. Press ? when your cursor is in the first position of the command-entry line (you should see a 1> at the start of this line). This HELP menu summarizes some of the information you have learned in this section. Press any key to return to your worksheet.

This HELP menu changes depending on what you are doing when you press **?**.You will see some of these changes when you start entering commands.

To leave SuperCalc and return to CP/M, type / Q.

The prompt line now reads:



```
EXIT SuperCalc? Y(es) or N(o)
```

If you press **Y**, you return to CP/M and see the A> prompt. The work you have done in this session is not saved.

If you press **N**, you return to SuperCalc and the worksheet you were working with remains on the screen.

If you want to continue with the intermediate level of this section, press N. If you want to stop the tutorial at this point, press Y.

Moving Around SuperCalc

Intermediate Level

More on Cursor Moving

More on Cursor Moving

There is another way to move the worksheet cursor to a new active cell. You can use the alternate diamond keys. Hold down the CTRL key while you press the S, D, E, or X key to move left, right, up, or down in that order.

^E	Move the worksheet cursor up
^X	Move the worksheet cursor down
^S	Move the worksheet cursor left
^D	Move the worksheet cursor right

These CTRL-key combinations form a compass on the keyboard and are the same in WordStar. They are available to the user who prefers controlling the cursor with the left hand.

Moving Around SuperCalc

Advanced Level

Using the Current Cell

Using the Current Cell

Whenever SuperCalc requires an address, you can use the active cell by typing the address of the active cell on the command-entry line, or by pressing the ESC key.

ESC ***Put the Current Cell Address on the Command-Entry Line***

When you press the **ESC** key, the SuperCalc program puts the location of the active cell onto the entry line for you to use in a command or expression.

After you press **ESC**, the arrow and alternate diamond keys control the worksheet cursor. If you move the worksheet cursor, the active cell address on the entry line reflects the new location. When you press **ESC** again, the address stops changing, and the arrow and diamond keys are again available for editing.

Typing a colon after the active-cell address "fixes" that address. The address before the colon is fixed; the address after the colon is still changeable.

The new active-cell location is temporary. When you press **RETURN** to enter the command or expression, the worksheet cursor returns to the prior active-cell location. If you were entering data into a cell, it would go into that prior location.

Entering and Deleting Data

Basic Level

This section explores basic concepts and commands of the SuperCalc program.

Editing the Command-Entry Line
Entering Data into Your Worksheet
Data Entry
Entering Text
Entering Numbers
Addressing a Cell
Entering Commands
The Insert Command
The Delete Command
The Blank Command
The Zap Command

To enter data into your worksheet, first move the worksheet cursor to the cell using the arrow keys (or the alternate diamond keys).

When your worksheet cursor is in the correct cell, type the value you want the cell to contain. This value will appear on the command-entry line.

Press RETURN to enter the value where your worksheet cursor is (this is the active cell).

We discussed moving the worksheet cursor in the previous section. In this section we will discuss entering and editing data.

Editing the Command-Entry Line

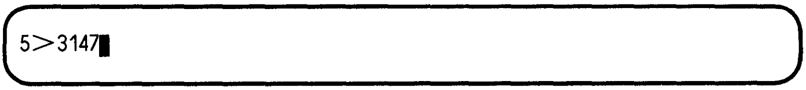
Once you begin to type data on the entry line, and before you press RETURN, the four arrow keys—and the alternate diamond keys—no longer move the worksheet cursor around the worksheet. Instead, you can use them to edit information on the entry line. You can always correct your commands or data while they are on the entry line.

Left and right arrow keys (and ^S or ^D) move the cursor on the command-entry line. If you make a mistake, move the cursor left or right with these keys, and then retype the entry. Use the SPACE BAR to erase any extra characters at the end of the line.

As you type on the command-entry line, the number at the start of the line tells you your current cursor position. For example, if you type

3147

And your cursor is after the 7, your command-entry line will look like this



```
5>3147█
```

Check this cursor position number to see whether you are exceeding the limits in data length described below.

If your cursor moves beyond position 76 on the command-entry line, the characters on this line will scroll to the left. Press the left-arrow key to re-examine characters which have scrolled off the screen.

Entering Data into the Worksheet

You enter data from the command-entry line into the worksheet by pressing RETURN. What you see on the entry line is what is entered into the active cell. SuperCalc enters all the data into the worksheet, not just the material to the left of the cursor. If you have any extra data at the end of the line, press the SPACE BAR to erase these characters before you press RETURN.

Data Entry

You can enter numbers, text, or formulas on the command-entry line. Numbers and text are described in this section. Formulas are discussed in the next section.

Entering Text

You can enter text into your worksheet by pressing the quotation mark (") and then typing the text. The quotation mark must be the first character in the command-entry line (i.e., you must type it when the start of the command-entry line reads 1>). Text can be up to 115 characters long. The quotation mark in column 1 is not counted in this, so when you put your cursor on the last character, the cursor position will be 116.

Once you have typed the quotation mark, to indicate that you are entering text, you can enter any character into the text.

Use the arrow keys to correct mistakes.

To enter the text into the worksheet, press RETURN.

Entering Numbers

You can enter numbers into your worksheet by typing any of the following characters in the first cursor position of the command-entry line:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, -, or a period (.).

A period (.) as the first character indicates a decimal number

A hyphen (-) as the first character indicate a negative number.

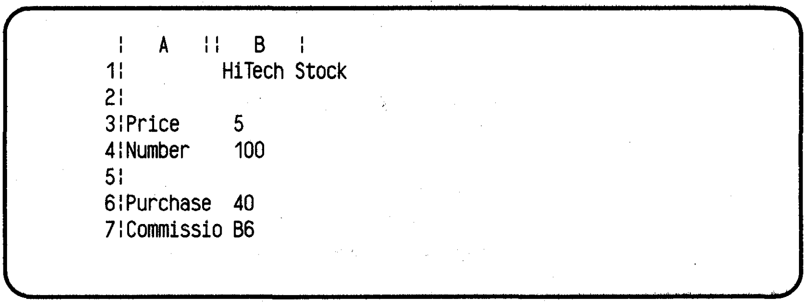
A plus sign (+) as the first character indicates a positive number.

If you type a number (0-9) as the first character, SuperCalc assumes the number is positive.

Pressing **?** after the first character displays a help screen about data entry.

Numbers can be up to 15 characters long. However, if you enter a number which is larger than the column width, SuperCalc will only display some of the number in the worksheet. This affects only the display on the worksheet, your entire number is remembered and will appear on the status line when you move the cursor to that cell. When you use this number in a formula, the entire number will be used.

Take the time now to learn to fill some blank cells. Enter some data and some text. (The rows and columns reflect what SuperCalc displays.)



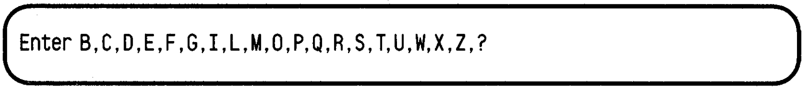
```
 ¦   A   ¦¦   B   ¦
1¦         HiTech Stock
2¦
3¦Price     5
4¦Number    100
5¦
6¦Purchase  40
7¦Commissio B6
```

Entering Commands

You can enter commands as well as data from the command-entry line.

The / Command Character

All SuperCalc commands begin by typing **/** as the first character in the command-entry line. When you press **/**, you see the following prompt:



```
Enter B,C,D,E,F,G,I,L,M,O,P,Q,R,S,T,U,W,X,Z,?
```

Press **?** for more information about these SuperCalc commands.

In this section we will discuss I(nsert), D(elete), B(lank) and Z(ap).

Entering the Command

After you have typed **/**, type the first letter of a SuperCalc command to display this command on the command-entry line. For example, to use the Insert command, press **I** after you press **/**. The command entry line automatically displays the entire command.

To make a correction, use the left or right arrow keys.

You can also press **^Z** to erase the entire line and start again.

Command Ranges

A few commands affect only the current cell, column, or row. Most allow you to specify which cell, column, or row is affected by the command. Type these locations on the command-entry line.

If you are addressing more than one cell, use a colon (:) to specify the range. This colon means address this command from the first cell to the last cell specified.

Examples:

C1	addresses a single cell
A1:A5	addresses a partial row from cell A1 through cell A5
A1:E5	addresses a block delimited by A1, A5, E1 and E5

You can type column addresses as either capital or lowercase letters. To use the current cell, column, or row you can press the comma (,) key.

You can edit addresses just as you do data by using the arrow keys.

The Insert Command

The Insert command inserts a new row or column of empty cells between existing rows or columns. A new row appears above the specified row; a new column appears to the left of the specified column.

/I *Insert a New Row or a Column*

The rest of the worksheet adjusts. Columns move right, rows move down. The contents of each column or row are preserved.

The contents, if any, of the last row (254) or column (BK) are discarded.

The command will not work if that last row or column contains a protected cell.

To use the Insert command, type / I. SuperCalc asks if you want to insert a

```
            R(ow) or C(olumn)?
```

If you press R (for R(ow)), SuperCalc asks:

```
            Enter Row Number
```

Type a number from 1 to 254, or type , for the current row. Then press RETURN.

If you reply C (for C(olumn)) to the first prompt, SuperCalc asks:

```
            Enter Column Letter
```

Type a column letter from A to BK, or type , for the current column. Then press RETURN.

The Delete Command

The Delete command deletes the contents of the specified row, column, or file. Deleting files is covered in a later section.

`/` `D` *Delete Data From a Row or Column*

Type `/` `D` to delete a row or column. SuperCalc asks if you want to delete:

```
R(ow), C(olumn) or F(ile)?
```

If you press `R` (for R(ow)), SuperCalc asks

```
Enter Row Number
```

Type a number from 1 to 254, or type `,` for the current row. Then press RETURN.

If you reply `C` (for C(olumn)), to the first prompt, SuperCalc asks

```
Enter Column Letter
```

Type a column letter from A or BK, or type `,` for the current column.

The command will not execute if there is a protected cell in that row or column.

When you delete a row, the rows below the deleted row move up, and all row numbering adjusts. If row 4 is deleted, row 5 moves up and becomes the new row 4, and so on.

When you delete a column, the columns to the right of the deleted column move left. If column D is deleted, column E moves and becomes column D, and so on.

The Blank Command

The Blank command deletes cells, partial columns, partial rows, or blocks.

/B *Blank the Contents of a Cell or Range of Cells*

When you type /B, for Blank, SuperCalc asks:

```
            Enter Range
```

Type the range you want Blanked. For example, to erase the contents of cell A1 through cell A5, type A1:A5. Protected cells will be bypassed.

The Zap Command

The Zap command erases the current worksheet and presents you with a new, blank worksheet. Unless you have saved the worksheet in a file, the contents are lost forever.

/Z *Clear the Entire Worksheet of Data*

Type /Z to Zap a worksheet. SuperCalc asks:

```
          Y(es) to clear everything, else N(o)
```

Press **Y** to erase your entire worksheet.

Press **N** if you don't want to erase the worksheet.

/Z is the only command that can override protection of cells.

Remember, when you Zap the worksheet, nothing remains.

Note:

Displaying data on your worksheet is covered briefly in the intermediate level of this section, and described in detail in the "Displaying" section. When you read about displaying data, keep in mind that some of the commands that erase data also erase the way the are displayed.

***Delete** erases the display format of the row or column deleted.*

***Blank** erases the display format of the cell, partial column, partial row, or block that is blanked.*

***Zap** erases the display format of the entire worksheet. When you Zap a worksheet, anything you did to that worksheet is erased.*

*You will also learn, in "Editing and Protecting," that you can protect data. **Insert, Delete,** and **Blank** will not erase protected cells. **Zap** is the only erasing command that will erase protected cells.*

Editing and Deleting Data

Intermediate Level

Editing the Command-Entry Line *Displaying Data*
Repeating Text *Entering Text*

Editing the Command-Entry Line

SuperCalc has additional commands to edit the command-entry line. For example, type in a text entry with the following incorrect spelling but do not press RETURN:

`` E p e n s e s

You could use the left-arrow key to backspace and retype from the point of the error. However, this time just move the cursor to p in Epenses. Notice that the number 3 appears at the left of your entry line to tell you that you're on the second character in the entry.

Press the ⌂ key. SuperCalc creates a space so you can insert a character. Type x and press RETURN.

⌂ ***Make Space for a Character.***

Move your worksheet cursor to another cell and type the following misspelled text. Don't press RETURN.

`` A i r r □ f a r e

To delete the extra "r", move your cursor below the second "r" in "Airr" and press the ⌦ key. The text to the right of the deleted character will move left one space to close up the space. Press RETURN.

⌦ ***Remove a Character.***

Enter some other samples. Practice with the editing keys until you are confident with this in-line editing feature. Try it with numbers, as well.

Note:

Regardless of where the cursor is on the entry line, all the visible text or numeric values go into the active cell when you press RETURN.

Entering Text

If the text you enter is too large for the column, your entry continues into the adjoining columns on the worksheet. If there is data in any of these columns, text extending into these columns is not displayed on the worksheet. The entire text entry is shown on the status line and retained in memory.

Repeating Text

You can repeat a text entry by typing ' instead of " as the first character on the command entry line. For example, move your worksheet cursor to cell A17, type ' # and press RETURN. Row 17 on your worksheet has a line of #'s across it.

The text will repeat, with no spaces between the entries, until it reaches a column with data in it. Repeating text is indicated on the status line as **Rtxt=**. If you typed **'#**, the status line reads **Rtxt='#**.

Displaying Data

There are several options available for displaying text and numbers on the worksheet. These options are discussed more fully in the "Displaying" section, but we will cover some of them briefly here.

Displaying numbers

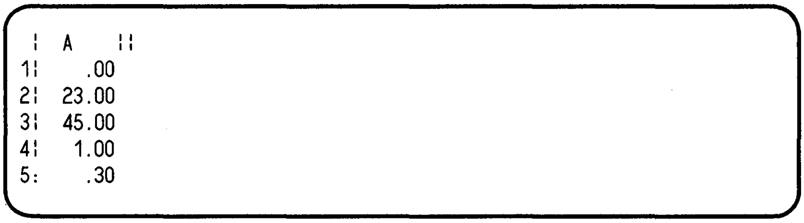
The way numbers appear when they're displayed on the worksheet depends on the display format you've selected, not on the way you type them. When you first start SuperCalc, numbers on the worksheet are displayed in General format. This format displays numbers as they best fit into the cell. If you are dealing with special numbers (such as integers only, or decimal numbers) you can choose to display the numbers in these special formats.

To change the way numbers are displayed, use the **Format** command. Type / F, for **Format**, on the command-entry line. We will make all the numbers in column A decimal numbers. Press C to change the display format of a column. When asked for the column letter, type A for column A and press RETURN. You are asked to "Define Formats". Press ? for a list of options. We want the **$** option for decimal numbers. Press any key to return to your worksheet, and then press $ and press RETURN. Now try typing some numbers into column A of your worksheet.

Move your worksheet cursor to cell A1 and type:

. RETURN
23 RETURN
45 RETURN
1 RETURN
.3 RETURN

Column A of your worksheet looks like this:



```
 |  A    ||
1|     .00
2|  23.00
3|  45.00
4|   1.00
5:     .30
```

SuperCalc has converted your entries into decimal numbers.

Displaying Text

You can also change the display format of text from left-justified text to right justified text by using the **Format** command.

Editing and Deleting Data

Advanced Level

Data Limits

Data Limits

Numbers can have:

For ordinary numbers: 16 significant digits, plus optional decimal point and optional sign.

Largest ordinary number: 9999999999999999

Smallest ordinary number: -9999999999999999

For exponential numbers: 16 significant digits plus decimal point and optional sign for exponential numbers (scientific notation). These 16 digits can be raised to the 62nd power of 10.

Largest exponential number: 9.999999999999999E62

Smallest exponential number: -9.999999999999999E62

Text can have 115 characters.

Formulas can have 116 characters and can include arithmetic expressions, relational expressions, functions, and references to cells.

Formulas

Basic Level

What is a Formula

Formulas specify calculations and comparisons. Formulas use numbers, values in other cells, operators, and built in functions. Some examples of formulas are

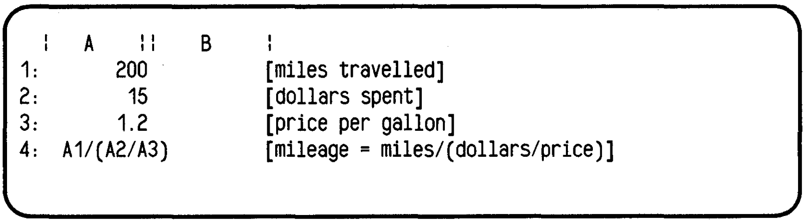
```
3 + 5
(A6 + 4) * B7
AVERAGE(3,5,A6)
```

Simple formulas may be used to average a row or sum a column of numbers; somewhat more complicated formulas may be used to balance ledgerbooks or evaluate expense accounts.

To enter a formula into a cell, move the worksheet cursor to the desired cell, type the formula on the command entry line, and press return. SuperCalc will evaluate the formula and write the value into the active cell. The formula still remains in memory, however. When the worksheet cursor is on this cell, the formula will be displayed on the status line.

Form=formula

The following example will show how a formula may be used. Suppose you want to determine what mileage your car is getting. You bought $15.00 worth of gas at $1.20 a gallon and travelled 200 miles. On your SuperCalc worksheet, enter the following data in Column A.



```
  |   A    ||    B    |
1:      200            [miles travelled]
2:       15            [dollars spent]
3:      1.2            [price per gallon]
4:  A1/(A2/A3)         [mileage = miles/(dollars/price)]
```

When you enter the formula in A4, remember that SuperCalc will display its value (in this case 16) in the cell. Suppose you make this trip in a different car, and you spend only $12.00 on gas. To determine the mileage of this car, simply change the entry in cell A2 to 12. SuperCalc will automatically adjust the value in cell A4 to reflect the new mileage (20). This is a very useful feature. Whenever new data is entered onto the worksheet, SuperCalc automatically recalculates any formulas which make use of that data.

Moving Referenced Data

If a formula references a cell (e.g., the formula 6*A2 references the cell A2 and uses information in this cell), and that cell is moved, SuperCalc automatically adjusts the formula to reflect the new location of the information. For example, if you enter the formula A1+A5 and then move row 5 to row 13 SuperCalc will automatically adjust the formula to A1+A13.

Formulas

Intermediate Level

Copying Formulas
Values or Formulas
Operators
Formula Limits

SuperCalc uses the most current value of a cell to evaluate a formula that references the cell. Formulas are evaluated from left to right; multiplication and division are performed before addition and subtraction. Terms within parentheses are evaluated first.

Copying Formulas

When a cell containing a formula is copied, SuperCalc normally adjusts all references in the copy of the formula to reflect the new location. (If, for example, you copy the formula A1+6 from cell A3 to cell B3, SuperCalc will adjust the copy of the formula to say B1+6.) If the **Delete, Insert** or **Move** commands are used to move a cell, SuperCalc will automatically make these adjustments. If the **Copy** or **Replicate** commands are used to make a copy of a cell, SuperCalc gives you the option of adjusting the entire formula, not adjusting the formula at all, or adjusting only certain referenced cells in the formula (Return, N(o Adjust, A(sk for Adjust, etc.).

Values or Formulas

Sometimes you will want SuperCalc to display the formula you have entered and not its value (e.g., you enter 2+6, and you want the worksheet to display 2+6 and not 8, which it would normally display). You can do this using the F(ormula) option of the **Global** command. You also have the option of retaining just the value and not the formula when you use **Load**, **Save**, **Replicate**, **Copy**, or **Output**.

Operators

Operators are the symbols used to manipulate numbers. SuperCalc recognizes the following operators:

+	(addition)	-	(subtraction)
*	(multiplication)	/	(division)
^	(raising to a power)		
=	(equals)	<>	(does not equal)
<	(less than)	<=	(less than or equal to)
>	(greater than)	=>	(greater than or equal to)

Formula Limits

Formulas can have up to 116 characters and may include numbers, operators, functions, and references to values in other cells.

Formulas

Advanced Level

Global
Deleting Referenced Cells
Forcing Recalculation (!)
Built-in Functions
NA

SuperCalc normally recalculates the worksheet every time you enter new data. When the program performs this recalculation, it proceeds by rows (i.e., it recalculates in the order row 1, row 2, row 3, etc.). You can, however, have SuperCalc recalculate by column using the C(olumn) option of the **Global** command:

```
/Global, C(olumn)
```

If you have a formula in cell B3 which depends on information in cell A20, you would want to use this option to force recalculation by column. This way, SuperCalc will read all the new information in column A before evaluating the formula in B3. To switch back to recalculation by Row, use Global, R(ow).

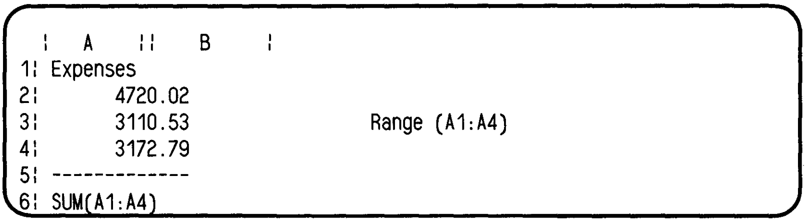
Forcing recalculation with !

SuperCalc may take a long time to go through all necessary recalculation of formulas. You can suspend automatic recalculation by typing **/GM** for Global Manual recalculation. Now SuperCalc will only recalculate formulas when you press !. Thus, manual mode allows you to make periodic recalculations at your convenience. To re-establish automatic recalculation, type **/GA** for Global Automatic recalculation.

Deleting referenced cells

If you delete a cell referenced by a formula, SuperCalc will display ERROR in the cell containing that formula. For example, if cell A3 contains the formula A1+6, and you delete row 1, then A3 will display ERROR. You must use the **Edit** command to rewrite the formula.

Deleting a cell from the middle of a range of cells referenced by a formula will not cause an error. Deleting a boundary cell from a range of cells will cause an ERROR, however, because SuperCalc will no longer know where the range should begin or end. For example, you may delete from the middle of the range A1:A12, but deleting A1 or A12 will cause an ERROR. A simple way to avoid this problem is to place text at range boundaries. By doing this, values will fall in the middle of the range and can be deleted or inserted as you wish.



```
  ¦  A   ¦¦    B    ¦
 1¦ Expenses
 2¦      4720.02
 3¦      3110.53                  Range (A1:A4)
 4¦      3172.79
 5¦ ------------
 6¦ SUM(A1:A4)
```

Both A1 and A5 are text, which has a numberical value of 0. Thus they will not affect your calculations. But now you can insert or delete any numerical value on the list without affecting the boundaries of the range and causing an ERROR.

NA (Not Available)

Sometimes you may be working on a complex worksheet with formulas that reference other cells. Until you enter all your data, SuperCalc will base calculations of formulas on empty cells (which SuperCalc treats as zeros) or old data. To remind yourself that the formulas do not reflect correct or current information, you can enter **NA** (for not available) into any cell which will need to be updated. Formulas which reference these cells will display N/A until the new data is entered. In effect, NA acts as a place holder, telling SuperCalc that you intend to put data in the cell at a future time.

The Built-in Functions

The built-in functions are listed in alphabetic order.

ABS(value)
Provides the absolute value.

AVERAGE (list)
Provides the arithmetic mean of the nonblank values in the list.

COUNT (list)
Returns the number of nonblank entries in the list.

ERROR, NA
Displays ERROR or NA (not available) for the cell having this function and for any cell with a formula referring to this cell.

EXP(value)
Raises "e" exponentially. The value is the exponent.

OR (expression 1, expression 2)
Results in "true" (1 is displayed on worksheet) if either expression 1 or expression 2 is nonzero. If both expressions have a value of 0 the function is "false", and 0 will be displayed on the worksheet.

AND (expression 1, expression 2)
Results in "true" (1 is displayed on worksheet) if both expression 1 and expression 2 are nonzero. If either expression has a value of zero, the function is "false", and 0 is displayed on the worksheet.

NOT (expression)
Results in "true" (1 is displayed on worksheet) if expression 1 has a value of zero. If the expression is nonzero, the function is "false" and O is displayed.

IF (exp1, exp2, exp3)
If expression 1 is nonzero, then use expression 2. If expression 1 has a value of zero, then use expression 3. Expression may be combined with AND, OR, or NOT to form expression 1.

INT (value)
Returns integer portion of value. The value is not rounded. Do not confuse this function with/FORMAT,I which will round off numerical entries.

LOOKUP (value, column/row range)
Searches the range for the last value less than or equal to the search value given. Returns the adjacent value from the column to the right of the search column of the row below the search row. Assumes the search range is in ascending order of values.

Ln (value), LOG 10 (value)
Provides the natural log, log base 10, of the value.

MAX (list), MIN (list)
Provides the maximum or minimum value in the list.

NPV (discount, column/row range)
Nets the present value of a group of cash returns at the given rate of discount. The cash amounts are assumed to be projected for equal time periods, such as every year; and the discount rate is for that interval. The first cash entry is discounted once, the second twice, and so forth, and added to form the total value.

PI:
Returns pi to 16 significant digits.

SIN (value, ASIN (value), COS (value)
ACOS (value), TAN (value), ATAN (value)
Trigonometric calculation of the value. ASIN is arcsine, etc. Trigonometric results are given in radians.

SQRT (value)
Returns the square root of the value.

SUM (list)
Returns the sum of the values in the list.

Here is a quick explanation of what "value," "range," "expression" and "list" mean in this context:

Value: a constant, the value of a cell, or a combination of these values made by using the arithmetic operators.

Range: any cell, partial or complete column or row, or block of cells. Example: AZ, A, A7:B:12

Expression: any number, range, or formula. Example: Z, A7:B12, SUM(A1:A7)

List: a list of expressions separated by commas. Example: (AZ, 65, B1:C3, 4*SUM(A1:A7)

Loading and Saving

Basic Level

These commands deal with transferring SuperCalc files, that is, fully developed worksheets, in and out of the SuperCalc program. They also cover file transfer to another device, such as a printer.

Loading a Worksheet

You can use the **/L** (or Load) command to read a SuperCalc data file from a diskette and load it into memory. The display reflects the evaluation of the contents of the file. You may load all or part of a worksheet.

/ L ***Load and Display a SuperCalc Worksheet.***

The prompt is:

```
Enter File Name (or <RETURN> for directory)
```

Enter the name of the file you would like to work with. The file name may have the .CAL extension. If you omit this extension, it is supplied; you do not have to type it. For example:

BALANCE<cr>

would load the file from drive A. On the diskette, its name will be BALANCE.CAL.

B:BALANCE<cr>

would load a file with the same name from the B disk drive.

If you press RETURN in response to this prompt, you are given the opportunity to choose the alternate disk drive (C), ask for directory of all files (D), or limit the list to SuperCalc files (S). ^Z returns you to the /L command.

After you type in a file name, you have a choice of loading the entire file or a specific portion of the file. The following prompt displays:

```
        A(ll) or P(art)?
```

If you reply **A**, the entire worksheet is loaded.

Loading P(art) is discussed in the Intermediate Level.

Naming Files

Here are the rules for naming your data files while using SuperCalc. (You need to know when to name a SuperCalc data file before you can properly save your worksheet data.)

B:STOCKRPT.CAL

B: is the drive designation
STOCKRPT is the actual filename
.CAL is the optional extension

A. You use a letter and colon to represent the drive that contains (or will contain) the disk file. If the drive designation is omitted, SuperCalc uses the A drive.

B. The actual file name can be from 1 to 8 characters long. It can contain numbers and letters, both upper and lower case. You can also use a few symbols (such as +), but it is best to avoid symbols since many special characters are not legal in file names. Do not use:

   < > . , ; : = ? * [ ]

   within a file name.

C. You may add the extension .CAL or omit it. If you omit it, SuperCalc will add it for you.

Saving a Worksheet

You can use the **/S** command to store the worksheet contents and all settings on a disk file.

/ S ***Saves an entire worksheet***

The prompt is:

```
Enter File Name (or <RETURN> for directory)
```

Enter the name you have chosen for saving your worksheet. Also enter the drive designation if you do not want to write it to the disk in the default drive (A). The SuperCalc program will automatically save the file with a .CAL file extension; you do not need to enter it as part of the file name.

In response to the prompt:

```
A(ll), V(alues) or P(art)?
```

type **A** to save the entire worksheet.

If you specify the name of an existing file, the program will display the following prompt:

```
File already exists:
C(hange name),B(ackup) or O(verwrite)?
```

The C option specifies that you would like to supply a different name. The B option tells SuperCalc to change the name of the disk file to filename.BAK and save the current worksheet under filename.CAL. The O option asks that the previous file contents be overwritten by the current worksheet.

^Z goes back up the command chain and, if issued after this warning message, gives you the opportunity to change your mind.

Printing a Worksheet

The **/O** command outputs part or all of the worksheet.

/ O ***Outputs to a printer or to a disk file for later printing***

Its prompt is:

```
D(isplay) or C(ontents) report?
```

D outputs the worksheet information in the way it is displayed.

C specifies that the output should be the actual contents of the cells, that is, the formulas that produce the displayed values.

When the following prompt appears:

```
Enter range
```

you may specify a single cell (e.g., B10), a row (e.g., 22), a column (e.g., BB), or a block of cells (e.g., A1:X9).

You may also enter: All

After you specify the portion of the worksheet to output, a prompt asks whether you want the data output to the printer, console (displays specified portion of the worksheet), or disk. The specific options will be covered in the Intermediate Level of this section.

Loading and Saving

Intermediate Level

Loading Part of a Worksheet
Saving Part of a Worksheet
Deleting a File
Controlling Printer Output

Loading Part of a Worksheet

The /L command can be used to load part of a worksheet. The specified cells can be loaded into the location you choose. Options let you adjust formulas and load values only.

When you see:

```
        A(ll) or P(art)?
```

reply **P**. Then further questions appear on the prompt line:

```
        From? (Enter Range)
```

Specify the range (i.e. A1:C8) of the saved worksheet that you wish to load and press RETURN.

```
        To? (Enter Cell), then RETURN, or '','' for Options.
```

Enter the cell address at the upper left of your destination. This will be a new location for that portion of the worksheet you are loading, and will replace the current cell contents.

Press RETURN if you wish automatic adjustment of formulas for the new location; otherwise type a comma for options. The options are: **N**(o Adjustment), **A**(sk for Adjust), or **V**(alues) only.

SuperCalc will prompt you with:

```
        N(o Adjust), A(sk for Adjust.), V(alues)
```

N copies formulas exactly as they are.

A allows you to choose for each reference to another cell address within a formula whether to copy it as is, or to have the SuperCalc program adjust it.

V copies the values only, without formulas.

When you choose the A option, SuperCalc asks about each formula: does it qualify for adjustment. The source and destination address of the formula are shown on the prompt line. SuperCalc positions the cursor at each cell reference on the entry line and asks you to reply Y or N. Y means yes, automatically adjust. N means no adjustment, transfer as is.

Note

If there are protected cells in the destination area, they will remain unchanged.

Saving Part of a Worksheet

The **/S** command can be used to save a portion of a worksheet to disk. When SuperCalc asks:

```
        A(ll), V(alues) or P(art)
```

A specifies that all cell contents will be saved.

V specifies that values will be saved without formulas.

For A and V, all of the following information is also saved: format settings, global options, title locking, window splitting, and active-cell location.

P allows you to save part of your current worksheet. When saving part of the worksheet, you can decide whether to save the entire portion (A) or just the values (V).

Deleting a File

The **/D** command can be used to delete a file. When SuperCalc asks:

```
        R(ow), C(olumn), or F(ile)?
```

Press F and the prompt will be:

```
Enter File Name (or <RETURN> for directory)
```

Pressing RETURN will give you the options of choosing an alternate disk drive and seeing the disk directories. Enter the drive the file is on and a colon, then the file name. Press RETURN.

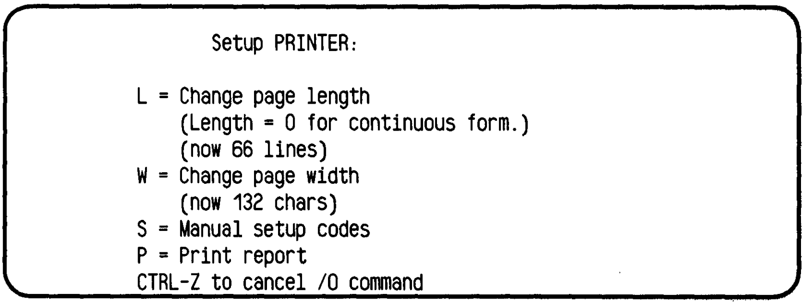
Controlling Printer Output

The **/O** command has options that allow you to change the default printer settings. At the fourth level, when you see:

```
Enter Device: P(rinter), S(etup), C(onsole), D(isk)
```

type P to send the data to the printer, C to display it on the screen, or D to send it to a disk file.

If you type S, for setup, the following options are provided:



```
                Setup PRINTER:

        L = Change page length
            (Length = 0 for continuous form.)
            (now 66 lines)
        W = Change page width
            (now 132 chars)
        S = Manual setup codes
        P = Print report
        CTRL-Z to cancel /O command
```

You may change one or more of the above parameters, then type P to output the report.

^Z can be used to cancel the /O command. If the worksheet is on its way to the printer, the printer will finish printing the current buffer of characters before it stops.

Loading and Saving

Advanced Level

Creating a Worksheet for WordStar

Creating a Worksheet for WordStar

The **/O** command can send a copy of a worksheet to the disk, a copy that may be used in a WordStar document. This technique is useful for designing or producing neat, well-organized, professional-looking reports. It is also used to create a SuperCalc command file.

At the fourth level of the /O command after you have chosen D(isplay) or C(ontents), and specified the output range, you will see the prompt:

```
Enter Device: P(rinter), S(etup), C(onsole), or D(isk)
```

Pressing D directs SuperCalc to write a file that is compatible with WordStar's file format.

Quick Tip

If you like, you can use an extension such as .DOC to indicate this version of the worksheet is in a document file.

Displaying

Basic Level

This section covers display formats: displaying the contents of cells and controlling the display on the screen.

Changing the Display Format

The **/F** command lets you depart from the usual display where numbers are right-justified, text is left- justified, and the column width is 9.

/ F ***Fit (or Format) the Numbers into Another Notation***

The prompt is:

```
Enter Level: G(lobal), C(olumn), R(ow), or E(ntry)
```

Specify the portion of the worksheet to be affected. G specifies the entire worksheet. C specifies a particular column which SuperCalc will ask for. R specifies a particular row. If you press E, you see "Enter Range," that is, you can specify a partial column or partial row. Using E to specify formatting at the cell level provides the highest priority of formatting (precedence of formats is listed under Advanced Level, near the end of this section).

If you specify row (R) or entry (E), SuperCalc prompts:

```
Define Formats: (I, G, E, $, R, L, TR, TL, *, D)
```

Here is a partial list that gives you the more frequently used display formats:

I (integer) displays numbers as integers (e.g., 5.5 would be displayed as 5).

G (general) displays a number as a real number (e.g., 61.994).

$ (dollars) displays a number as a dollar amount (e.g., 11.52).

Quick Tip

A complete list of formats is given at the Intermediate Level, Special Formats.

Looking at Formulas

The primary use of the /G command is to alternate between displaying formulas and the values that result from evaluating formulas.

/ G ***Change Global Features***

The prompt is:

```
F(orm, N(ext), B(order), T(ab), R(ow), C(ol), M(an), A(uto)?
```

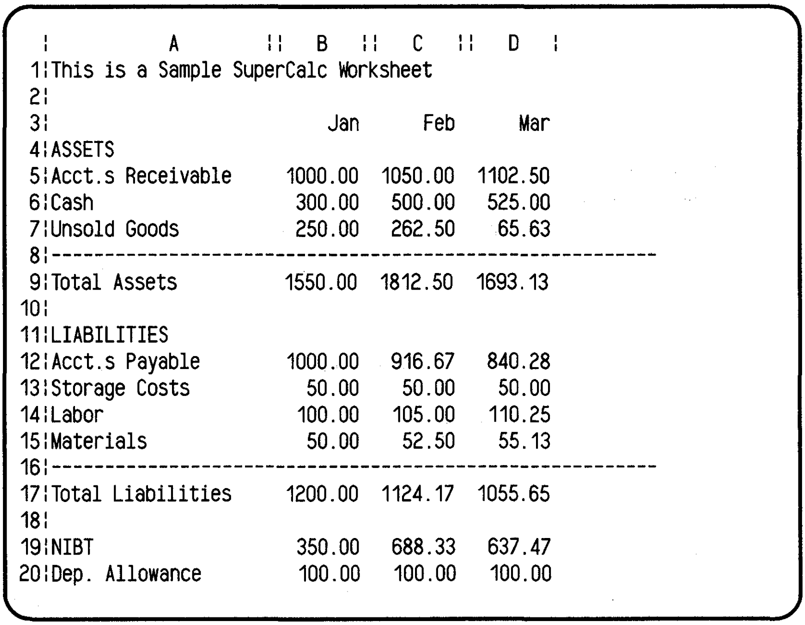
If you respond to the prompt by pressing F, the display will show the formulas contained in the cells instead of the values that result from the formula calculations. F is a toggle switch. If formulas are currently being displayed, pressing F will display the values.

Note

The remaining options are explained under Advancing Automatically and Forcing Recalculation in the Intermediate and Advanced section of Displaying.

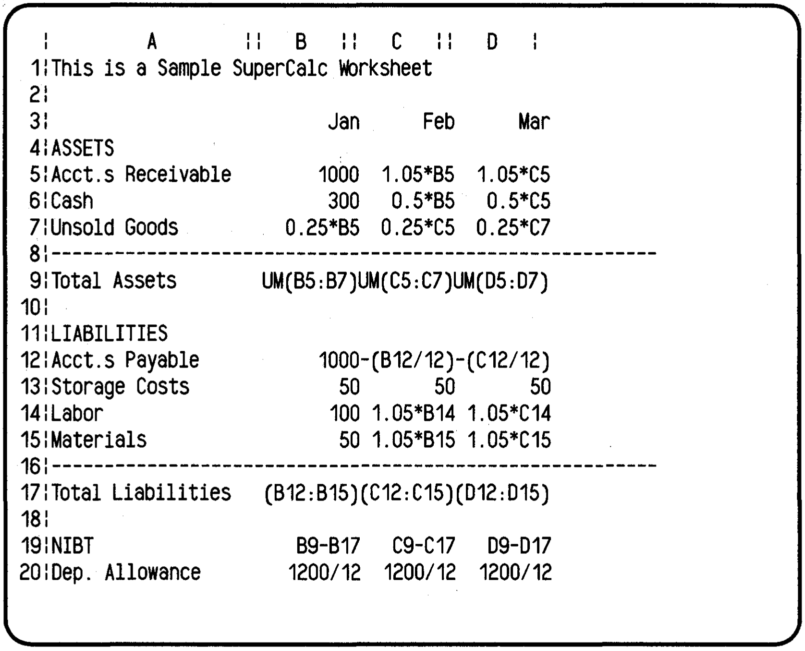
If you load the sample file called BALANCE, you can experiment with using /G and F to view the values and the formulas that are behind them.

Values:



```
  ¦          A        ¦¦  B  ¦¦  C  ¦¦  D  ¦
 1¦This is a Sample SuperCalc Worksheet
 2¦
 3¦                        Jan      Feb      Mar
 4¦ASSETS
 5¦Acct.s Receivable   1000.00  1050.00  1102.50
 6¦Cash                 300.00   500.00   525.00
 7¦Unsold Goods         250.00   262.50    65.63
 8¦---------------------------------------------------
 9¦Total Assets        1550.00  1812.50  1693.13
10¦
11¦LIABILITIES
12¦Acct.s Payable      1000.00   916.67   840.28
13¦Storage Costs         50.00    50.00    50.00
14¦Labor                100.00   105.00   110.25
15¦Materials             50.00    52.50    55.13
16¦---------------------------------------------------
17¦Total Liabilities   1200.00  1124.17  1055.65
18¦
19¦NIBT                 350.00   688.33   637.47
20¦Dep. Allowance       100.00   100.00   100.00
```

Formulas



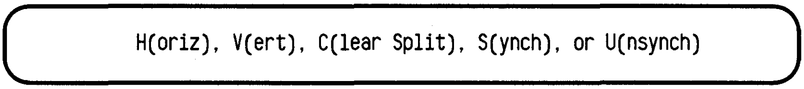
```
  ¦         A         ¦¦   B   ¦¦   C   ¦¦   D   ¦
 1¦This is a Sample SuperCalc Worksheet
 2¦
 3¦                           Jan      Feb      Mar
 4¦ASSETS
 5¦Acct.s Receivable         1000  1.05*B5  1.05*C5
 6¦Cash                       300   0.5*B5   0.5*C5
 7¦Unsold Goods           0.25*B5  0.25*C5  0.25*C7
 8¦----------------------------------------------------------
 9¦Total Assets         UM(B5:B7)UM(C5:C7)UM(D5:D7)
10¦
11¦LIABILITIES
12¦Acct.s Payable            1000-(B12/12)-(C12/12)
13¦Storage Costs               50       50       50
14¦Labor                      100 1.05*B14 1.05*C14
15¦Materials                   50 1.05*B15 1.05*C15
16¦----------------------------------------------------------
17¦Total Liabilities    (B12:B15)(C12:C15)(D12:D15)
18¦
19¦NIBT                    B9-B17   C9-C17   D9-D17
20¦Dep. Allowance         1200/12  1200/12  1200/12
```

Using Windows

You can view two widely separated areas of your worksheet simultaneously; for example, the text in column A and the formulas in column Z.

[/] [W] *Split the Worksheet into Windows*

The prompt is:



```
H(oriz), V(ert), C(lear Split), S(ynch), or U(nsynch)
```

H splits the screen horizontally. The current row moves down and a second border appears above it. The active cell moves up one cell in its column.

V splits the screen vertically. The current column moves right and a second border replaces it. The active cell moves left one cell in its row.

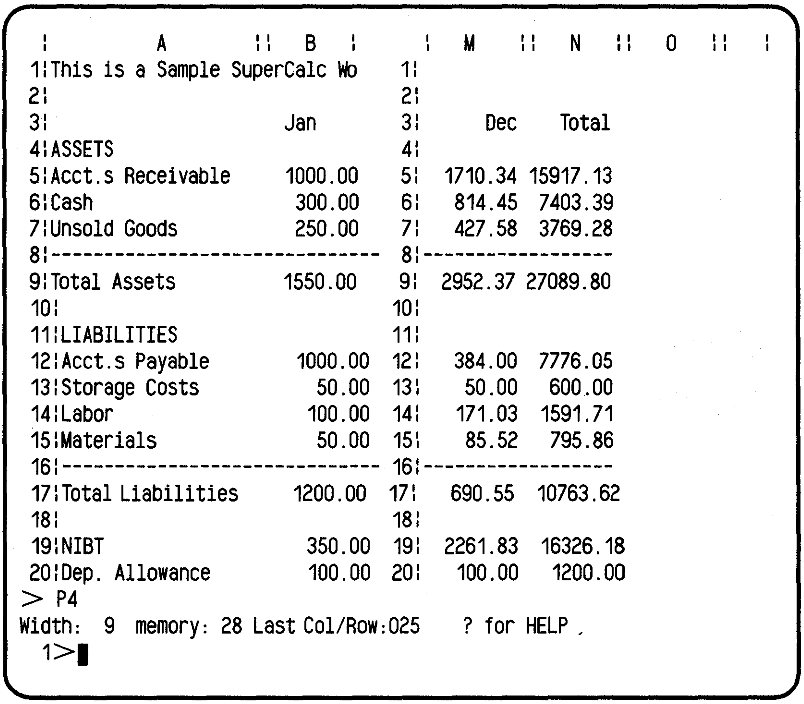
C clears the split screen. The portion that was above or to the left is the primary screen. SuperCalc now displays the entire primary screen.

Within the two portions of the screen, you can set formatting (/F) and global (/G) options independently. It is possible to show the same data with different formatting and options—for example, to show the same column as values and as formulas.

When the split is cleared, the options and formats for the primary screen remain. The primary screen is the portion above or to the left.

When a worksheet is split into two windows, you have two complete copies of the worksheet. You can think of these copies as being pasted on cylinders. You can rotate one cylinder or both.

The illustration of a vertical split shows columns A and B and a window that contains only column M. On your screen, they appear side by side, with a vertical line between them.



```
 ¦         A       ¦¦  B  ¦     ¦  M    ¦¦   N   ¦¦   O   ¦¦   ¦
 1¦This is a Sample SuperCalc Wo   1¦
 2¦                                2¦
 3¦                   Jan          3¦     Dec     Total
 4¦ASSETS                          4¦
 5¦Acct.s Receivable   1000.00     5¦  1710.34 15917.13
 6¦Cash                 300.00     6¦   814.45  7403.39
 7¦Unsold Goods         250.00     7¦   427.58  3769.28
 8¦------------------------------  8¦------------------
 9¦Total Assets        1550.00     9¦  2952.37 27089.80
 10¦                              10¦
 11¦LIABILITIES                   11¦
 12¦Acct.s Payable      1000.00   12¦   384.00  7776.05
 13¦Storage Costs         50.00   13¦    50.00   600.00
 14¦Labor                100.00   14¦   171.03  1591.71
 15¦Materials             50.00   15¦    85.52   795.86
 16¦----------------------------- 16¦------------------
 17¦Total Liabilities   1200.00   17¦   690.55 10763.62
 18¦                              18¦
 19¦NIBT                 350.00   19¦  2261.83 16326.18
 20¦Dep. Allowance       100.00   20¦   100.00  1200.00
> P4
Width:  9  memory: 28 Last Col/Row:025    ? for HELP ,
  1>█
```

To put the worksheet cursor on the other cylinder, type a semicolon (;).

Quick Tip

If SuperCalc does not appear to be accepting your split instruction, try clearing the previous specifications. Press C for Clear Split.

Displaying

Intermediate Level

Special Formats
Advancing Automatically
Synchronizing Window Display

Special Formats

If you specified global (G) or column (C) formatting under the **/F** command, SuperCalc prompts:

```
        Define Formats:  (I,G,E,$,R,L,TR,TL,*,D, column width)
```

For row (R) and entry (E), SuperCalc does not display "column width" because it is not relevant.

Here is a list of the display formats:

I displays numbers as integers. This rounds decimal fractions and converts them to whole numbers.

G (general) displays the number as an ordinary number if it fits in the column width; otherwise, it displays the number as an exponential number.

E (exponential) displays the number as an ordinary number in scientific notation as a power of 10. For example: 1776 is 1.776e3, 1,000,000 if 1e6.

$ (dollar amount) rounds to the nearest cent and appends .00 to whole numbers. No dollar sign is displayed.

R	right-justifies numbers
L	left-justifies numbers
TR	right-justifies text
TL	left-justifies text
*	(graphic display for numbers) uses asterisks to show the relative sizes of numbers. Allows bar graph display.
D	(default) resets to the next level of formatting.
	0-127 is the legal column width range for the specified column or for the worksheet.

When your entries are contradictory, the SuperCalc program will act on the one entered last. For example, if you enter R,L,I,G, then L and G will take effect, and SuperCalc will override R and I.

Note:

Format does not apply to data entry. The contents of a cell remain as entered; format specifies how the contents are displayed.

Advancing Automatically

In addition to letting you view the formulas on which values are based, the **/G** command lets you enter data in an automatic mode. You can turn off border display, auto-advance, and tabbing. (You will want the border display on when you enter values by individual cell address.)

The prompt at the second level is:

```
F(orm), N(ext), B(order), T(ab), R(ow), C(ol), M(an), A(uto)?
```

N causes the cursor to advance automatically in the "current direction" to the next cell after the data is entered in the current cell. If auto-advance is already in effect, pressing N turns off auto-advance; the cursor stays under the current cell.

B supresses the display of the worksheet border. If you already suppressed the border display, pressing B restores the border display. "Border" refers to the column and row designations across the top and down the left side of your display window.

T activates the Tab mode. In the Tab mode, advancing between cells skips all empty or protected cells. therefore, you can never select a protected or an empty cell as the active cell in this mode. T deactivates it if SuperCalc is already in the Tab mode.

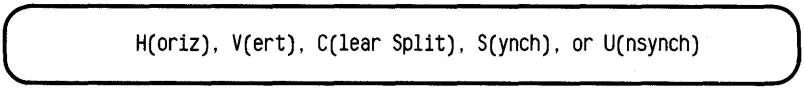
N, B, and T are all toggle switches.

Options R, C, M, and A concern recalculation.

Synchronizing Window Display

After you have split the worksheet into two areas, you can scroll through one side while the other stays fixed on the screen.

After you have specified the type of window (horizontal or vertical) under the /W command, type /W once again. You will see:

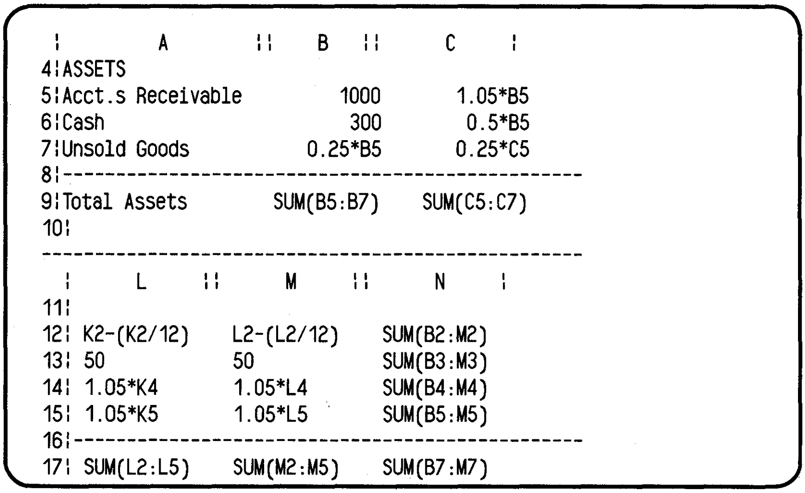


```
H(oriz), V(ert), C(lear Split), S(ynch), or U(nsynch)
```

S synchronizes the scrolling of the two portions.

U unsynchronizes scrolling; the two portions will scroll independently. The worksheet cursor resides in the portion that will respond to cursor-movement commands.

This worksheet illustrates a window display that is split horizontally. The display is also unsynchronized: part of the columns L, M, and N are scrolled under columns A, B, and C. This is a partial view of the screen:



```
  ¦        A       ¦¦    B   ¦¦      C     ¦
 4¦ASSETS
 5¦Acct.s Receivable        1000       1.05*B5
 6¦Cash                      300        0.5*B5
 7¦Unsold Goods          0.25*B5       0.25*C5
 8¦-----------------------------------------------
 9¦Total Assets       SUM(B5:B7)    SUM(C5:C7)
10¦
--------------------------------------------------
   ¦      L      ¦¦      M      ¦¦      N      ¦
11¦
12¦ K2-(K2/12)    L2-(L2/12)    SUM(B2:M2)
13¦ 50            50            SUM(B3:M3)
14¦ 1.05*K4       1.05*L4       SUM(B4:M4)
15¦ 1.05*K5       1.05*L5       SUM(B5:M5)
16¦-----------------------------------------------
17¦ SUM(L2:L5)    SUM(M2:M5)    SUM(B7:M7)
```

Note:

The semicolon (;) sends the worksheet cursor from one window to the other.

Displaying

Advanced Level

Precedence of Formats
Forcing Recalculation

Precedence of Formats

The **/F** command affects the format for a given portion of the worksheet. Options are G(lobal), by specific C(olumn), R(ow), or E(ntry).

The display formats can be specified as I(nteger) notation, G(eneral format), E(xponential) notation, $ dollar format, R(ight) or L(eft) justification, * asterisk-fill reflecting value. The D(efault) is the general display, numbers right-justified, text left-justified, and column width 9.

When you specify different formats for the same cell (perhaps a columnar specification conflicts with a row specification), the order of precedence is first the cell (E), then row (R), column (C), and finally worksheet or global (G). That is, cell formatting overrides any format for the column or row where the cell is. Where row and column intersect, row formatting overrides. All of these override the global settings.

Forcing Recalculation

A sophisticated use of the **/G** command is to manually control recalculation.

After you type $\boxed{/}\boxed{G}$, you will see:

```
F(orm), N(ext), B(order), T(ab), R(ow), C(ol.), M(an.), A(uto)?
```

R recalculates by rows, from the top down. (Rows are recalculated left to right.

C recalculates by columns, from the left across. (Columns are recalculated from top down.)

M sets recalculation as a manual function, to occur on request, in response to the ! key.

A sets recalculation as an automatic function (default).

Quick Tip

The ! command causes recalculation in the manual-calculation mode. In the automatic mode, it provides an additional recalculation.

Editing and Protecting

Basic Level

This section covers the commands that affect information once it is already in the cells of the worksheet.

Replacing Information

When a cell contains erroneous data, you can retype its entire contents. Alternatively, you can use the **/E** command to edit the contents of a specified cell, then place them back in the active cell. The /E command transfers cell contents to the command entry line for editing.

/ E ***Edit the Previous Contents of a Cell***

Its prompt is:

```
        From? Enter cell
```

Specify a cell in response to the prompt. To indicate the current or active cell type a comma (,). The cell contents come to the entry line, replacing the command on the line.

Once the contents are displayed on the command line, you can edit them with the pre-entry edit function. Use the arrow or diamond keys to move the cursor nondestructively left and right to characters you want to change. The character that will be altered is the one above the cursor.

You can also replace characters one-for-one by simply typing new characters over them.

You can delete characters, including blanks, by pressing the DOWN-ARROW key (or ^X). You can insert blanks by pressing the UP-ARROW (or ^E). Then if you wish, you can replace the blanks by typing other characters over them.

For example, if the active cell contains

Janaurry

type /E and a comma to bring this example to the entry line. Use the left arrow to move the cursor to the second "a" in Janaurry and type ua. Move the cursor right to one of the "r"s in Januarry, then press the down arrow to delete it, and press RETURN. (Remember, pressing RETURN puts the entire entry into the cell no matter where the cursor is positioned.)

The active cell now contains:

January.

Note

If the active cell is protected, you can not edit its contents.

Editing and Protecting

Intermediate Level

Protecting a Cell
Unprotecting a Cell

Protecting a Cell

The contents of some cells can be protected from the effects of commands operating over entire rows, columns, or blocks of data. The **/P** command shields the contents and formatting of specified cells from alteration. You can not enter or edit data in protected cells.

/ P ***Protects the Contents of a Cell from Alteration***

Its prompt is:

```
Enter Range
```

As usual, you can specify a single cell, an entire row, an entire column, part of a row or column (e.g., a2:a4), or a block of cells (e.g., specify the diagonal as in a1:z9). RETURN specifies protection for the current cell. ALL protects the entire worksheet.

Protection secures the contents of a cell against the effects of these commands:

/B, /D, /E, and against some of the effects of /F

/D will not delete any cells in the specified row or column if a protected cell is in that row or column.

Quick Tip

/Z ignores protection and alters the entire worksheet.

Unprotecting a Cell

You can return a cell's status to its initial state so that it can again be operated on by the /B, /F, /C, /R, and /L commands.

/ U ***Removes the Protection Created by the /P Command.***

Its prompt is:

```
        Enter Range
```

The range definitions are the same as for Protect.

Quick Tip

There is no error if you try to remove protection from something that is not protected.

Editing and Protecting

Advanced Level

Locking Titles into place

Locking Titles into Place

You can keep any portion of the worksheet locked in place while you scroll the rest of the screen. Since it is most often used for the title line, the **/T** command is called the title lock.

/ T ***Prevents a Row or a Column from Scrolling***

/T allows you to lock columns or rows into place on the display window. Locked information will not scroll; however, other information on the screen can scroll.

SuperCalc uses the current position of the worksheet cursor and locks the cells above it on the vertical axis or left of it on the horizontal axis.

Its prompt is:

```
H(oriz), V(ert), B(oth), or C(lear)?
```

H locks the current row and all rows above it.

V locks the current column and all columns to the left of it.

B locks both the current row and column and all rows above and columns to the left.

C removes the lock.

Working with Larger Blocks

Basic Level

Copying a Cell
Moving a Row or Column
Replicating a Cell

Copying a Cell

The **/C** command lets you copy the contents of a cell to a new location.

/ C ***Copy a Cell***

Its prompt is:

```
From? (Enter Range)
```

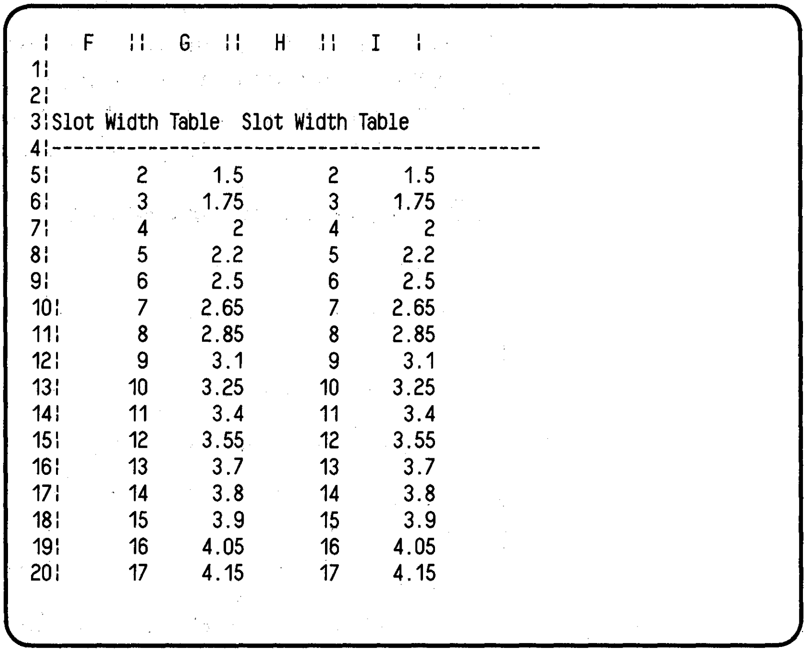
After you enter the address of a cell, the next prompt is:

```
To? (Enter Cell), then Return; or '','' for Options
```

Enter the address of the destination cell.

SuperCalc copies the contents of the first cell into the second and automatically adjusts any internal references to another cell.

Here is the BARRIER worksheet, one of the sample files. After copying columns F and G to H and I, the slot/width table appears in the file twice:



```
  ¦   F    ¦¦   G    ¦¦   H    ¦¦   I    ¦
1¦
2¦
3¦Slot Width Table  Slot Width Table
4¦----------------------------------------------
5¦         2      1.5         2      1.5
6¦         3     1.75         3     1.75
7¦         4        2         4        2
8¦         5      2.2         5      2.2
9¦         6      2.5         6      2.5
10¦        7     2.65         7     2.65
11¦        8     2.85         8     2.85
12¦        9      3.1         9      3.1
13¦       10     3.25        10     3.25
14¦       11      3.4        11      3.4
15¦       12     3.55        12     3.55
16¦       13      3.7        13      3.7
17¦       14      3.8        14      3.8
18¦       15      3.9        15      3.9
19¦       16     4.05        16     4.05
20¦       17     4.15        17     4.15
```

Quick Tip

Instead of entering a destination address, you can position the worksheet cursor at the first cell to receive the copied information. Pressing RETURN in response to To? uses the cursor's position as the current destination.

Moving a Row or Column

You can use the **/M** command to adjust the worksheet without destroying any data or performing any formatting. /M moves a column left or right and inserts it in a new location, or moves a row up or down and inserts it in a new location.

/ M *Move a Row or Column*

Its prompt is:

```
R(ow) or C(olumn)?
```

If your reply is R, the prompt is:

```
From? Enter row number
```

You may enter a number from 1 to 254.

If you reply C, the prompt is:

```
From? Enter Column Letter
```

You may enter a column designation from A to BK.

After you have specified a row or column, SuperCalc will ask To?, the destination of the move. Reply with a row or column designation, whichever is appropriate.

The /M command transfers the contents from one column or row to another.

All formulas on the worksheet adjust automatically as necessary. The adjustments preserve reference to cell contents by giving their new location.

For example, if a cell uses B12 is its formula and B12 moves to B4, after the move the formula is adjusted to reference B4. (B12*1.25) -> (B4*1.25)

Replicating a Cell

Like getting several reprints from a single photograph, you can replicate the contents of a cell into many cells by using the **/R** command.

/ R *Replicate the Contents of a Cell*

Its prompt is:

```
From? (Enter Range)
```

After you enter the address of a cell, the next prompt is:

```
To? (Enter Range), then Return; or '','' for Options
```

Enter the address or a range of addresses of the destination cells.

SuperCalc replicates the contents of the first cell into the second and automatically adjusts any internal references to another cell.

Quick Tip

Instead of entering a destination address, you can position the worksheet cursor at the first cell to receive the replicated information. Pressing RETURN in response to To? uses the cursor's position as the current destination.

For example,

/R, b12,e3:e8

replicates a cell into a partial column.

/R, b12,e3:j3

replicates a cell into a partial row.

/R, b3:b7, d3:j

replicates a partial column into a group of partial columns. In this example, the partial column is five cells deep. The result will be a block of cells repeating that partial column seven times. The top of that block is on row 3.

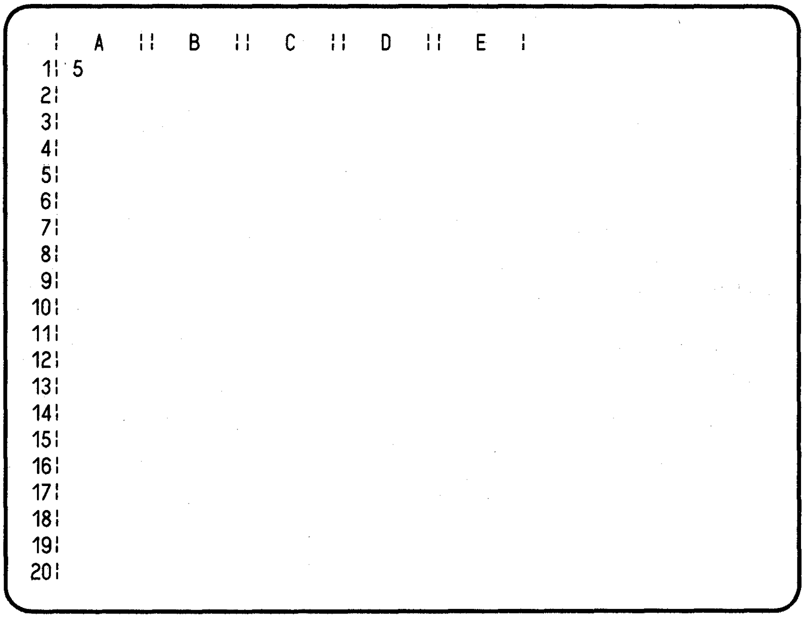
/R, b3:e3, g5:g7

replicates a partial row into a group of partial rows. The partial row here is four cells across. The result will be a block of cells repeating the partial row three times. The left side of that block is column G.

Note:

Both the /R and /C commands can make a one-for-one copy. /C can not make multiple copies. /R can not copy a block.

If a worksheet contains this data:

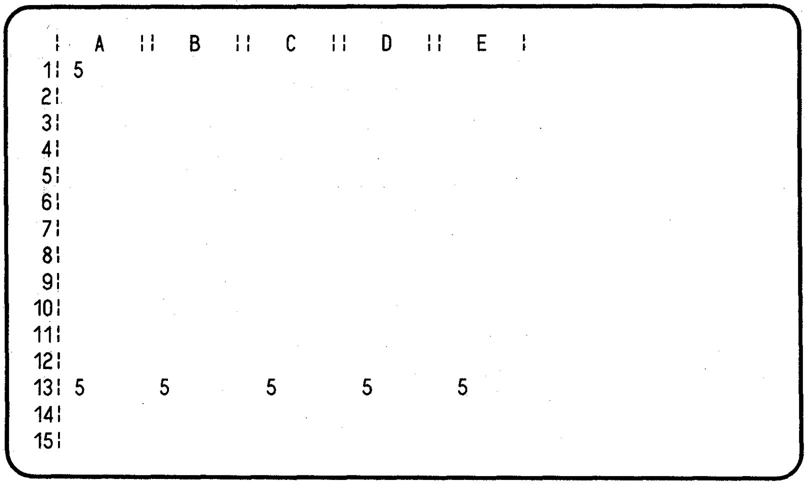


```
  ¦  A  ¦¦  B  ¦¦  C  ¦¦  D  ¦¦  E  ¦
 1¦ 5
 2¦
 3¦
 4¦
 5¦
 6¦
 7¦
 8¦
 9¦
10¦
11¦
12¦
13¦
14¦
15¦
16¦
17¦
18¦
19¦
20¦
```

The command

/R,al,al3:el3

creates this worksheet:



```
  ¦  A  ¦¦  B  ¦¦  C  ¦¦  D  ¦¦  E  ¦
 1¦ 5
 2¦
 3¦
 4¦
 5¦
 6¦
 7¦
 8¦
 9¦
10¦
11¦
12¦
13¦ 5       5        5        5        5
14¦
15¦
```

Working With Larger Blocks

Intermediate Level

Copying a Range of Cells

Copying a Range of Cells

In addition to copying a single cell, the /C command lets you copy to a new location, part of a column or row, or a block.

/ C ***Copy Contents to a New Location***

Its prompt is:

```
From? (Enter Range)
```

Specify a cell, a partial column, a partial row, or block.

The next prompt is:

```
To? (Enter Cell), the Return; or ‘‘,’’ for Options
```

Enter a single cell address to give the new location:

For a partial column, give the upper cell.
For a partial row, give the left cell.
For a block, give the upper left cell.

After you have entered the single cell address, press RETURN.

Copy makes a one-to-one copy of the source into a destination of the same shape and size. All the formulas will be copied and automatically adjusted; that is, all references to other cells are adjusted for their new location, if possible.

The command for copying part of a row looks like this:

```
/C, B9:B15, K20
```

The command for copying a block looks like this:

```
/C, B9:G15, K9
```

If you wish a choice of options for copying formulas, type a comma.

Working With Larger Blocks

Advanced Level

Copying Without Adjusting	*Working in Batch Mode*
Replicating Without Adjusting	*Creating a Command File*

Copying Without Adjusting

A more sophisticated use of the **command is control of the otherwise automatic adjustments.**
After you type /C, move the worksheet cursor to the first cell of the destination. Enter the address of the cells to copy from.

In response to the next prompt,

```
To (Enter Cell), then Return; or '','' for Options
```

Press comma to display this prompt:

```
N(o Adjust), A(sk for Adjust), V(alues)
```

N proceeds with the copy and does not adjust the formulas.

A asks about each cell in the range of the copy. SuperCalc displays the contents on the command-entry line and highlights the expression that will be adjusted if you answer Y to the prompt: Adjust (Y or N)

V copies only the values from the source cells to the destination.

For example:

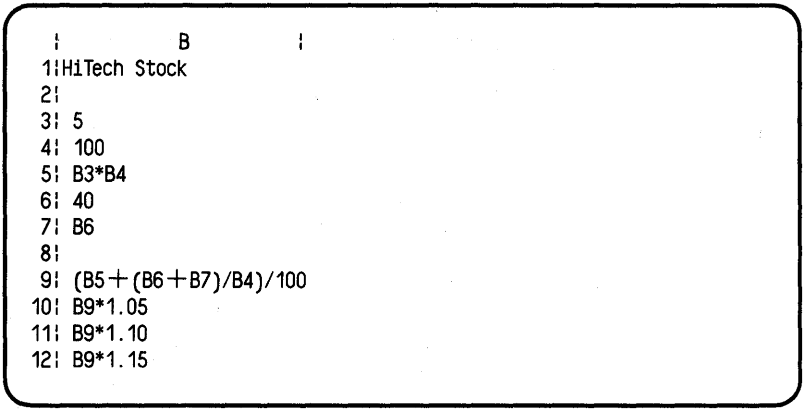
/C,b9,c12,N

copies without adjustment.

/C,b9:b15,e9,A

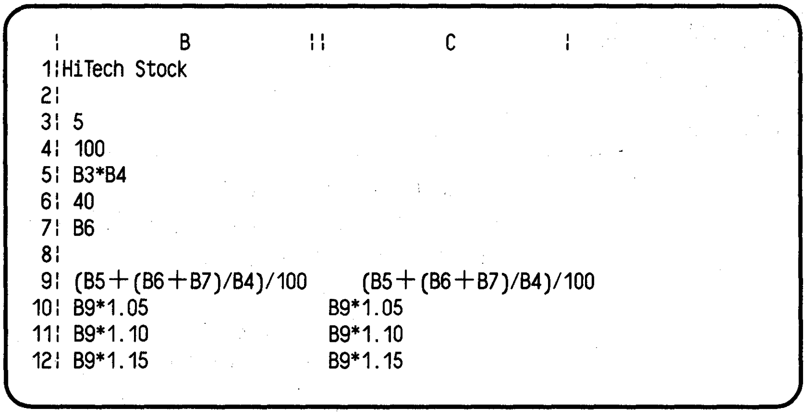
copies and asks whether or not to make the individual adjustments.

Here is an example that shows a worksheet that is copied with No Adjustment indicated. Starting with these values:



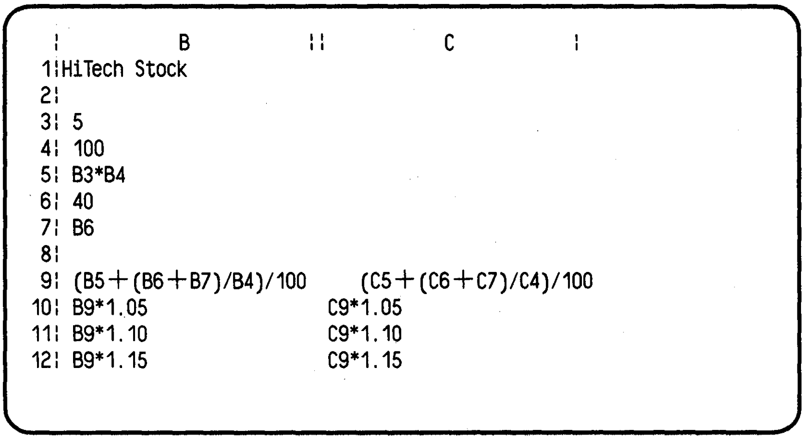
```
  ¦           B         ¦
 1¦HiTech Stock
 2¦
 3¦ 5
 4¦ 100
 5¦ B3*B4
 6¦ 40
 7¦ B6
 8¦
 9¦ (B5+(B6+B7)/B4)/100
10¦ B9*1.05
11¦ B9*1.10
12¦ B9*1.15
```

the next column is filled with unadjusted formulas:



```
  ¦           B         ¦¦            C         ¦
 1¦HiTech Stock
 2¦
 3¦ 5
 4¦ 100
 5¦ B3*B4
 6¦ 40
 7¦ B6
 8¦
 9¦ (B5+(B6+B7)/B4)/100       (B5+(B6+B7)/B4)/100
10¦ B9*1.05                B9*1.05
11¦ B9*1.10                B9*1.10
12¦ B9*1.15                B9*1.15
```

The next version illustrates adjusted formulas:



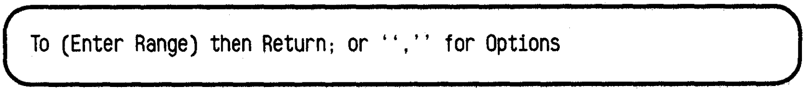
```
  ¦          B          ¦¦          C          ¦
 1¦HiTech Stock
 2¦
 3¦ 5
 4¦ 100
 5¦ B3*B4
 6¦ 40
 7¦ B6
 8¦
 9¦ (B5+(B6+B7)/B4)/100    (C5+(C6+C7)/C4)/100
10¦ B9*1.05               C9*1.05
11¦ B9*1.10               C9*1.10
12¦ B9*1.15               C9*1.15
```

Replicating Without Adjusting

In addition to replicating one cell into many, the **/R** command gives you control over otherwise automatic adjustments and replicating values only.

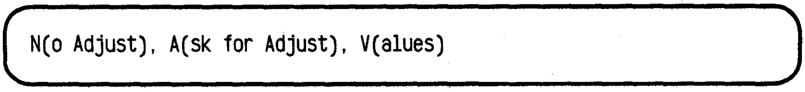
After you type /R, move the worksheet cursor to the first cell of the destination. Enter the address of the cells to copy from.

In response to the next prompt:



```
To (Enter Range) then Return; or '','' for Options
```

Pressing comma (,) displays the cursor's location and the prompt:



```
N(o Adjust), A(sk for Adjust), V(alues)
```

N proceeds with the replication and does not adjust the formulas.

A asks about each cell in the range of the replication. SuperCalc displays the contents on the command-entry line and highlights the expression that will be adjusted if you answer Y to the prompt: Adjust (Y or N)

V copies only the values from the source cells to the destination.

For example,

/R,b12,e3:j3,N

replicates without adjustment.

/R,b12,e3:j3,A

replicates and asks whether or not to make the individual adjustments.

Working in Batch Mode

A command file contains SuperCalc commands that are composed of characters exactly as you would enter them from the keyboard. The **/X** command lets you execute a command file.

/ X *Execute Commands from a Command File*

When you enter /X, the prompt line changes to:

```
Enter File Name (or <RETURN> for directory)
```

If you enter a file name, the SuperCalc program reads each of the commands in the specified file and executes that command, a character at a time.

If the file is not in the proper format or a command is in error, an error message is displayed on the status line and the X(ecute) command is abandoned.

Return puts you on the level where you can specify the other disk drive, display the disk directory of all files, or limit the list to SuperCalc files.

For example, to execute the command file MYEXEC.PRN, enter SuperCalc and type:

```
/X MYEXEC.PRN
```

Quick Tip

You can terminate the /X command at any time by pressing ^Z.

Creating a Command File

You can create a command file in SuperCalc or in WordStar.

To build a command file in SuperCalc, start with a clean worksheet (/Z). Next, eliminate the border with the Global command (/GB) so that no "frame" information will later be saved with the program. Then widen column A to 50 characters with the Format command (/FCA <RETURN> 50 <RETURN>), so you'll have room for long command strings on your worksheet.

Enter the command strings as a series of text entries. For example:

″/LSTOCKS,A	to load a SuperCalc file named STOCKS. Use a comma to signal the end of the file name; answer A(ll) or P(art) query. Continue with the commands you want.
″=A19	Goes to cell A19.
″″′New text	Adds a new title line.
″/SSTOCKS,O,A,	Saves STOCKS and overwrites existing file.

Use the /O command and a filename extension of .PRN, for example:

/O	Summons the Output command.
d	Answers D(isplay) or C(ontents) Report?
all <RETURN>	Answers All to Enter Range.
d	Requests D(isk) output.
MYEXEC.PRN <RETURN>	Names the file and gives it the necessary .PRN extension.

Similarly, in WordStar, to create a command file, start by creating a Nondocument file; type N. Use whatever WordStar commands you need to enter and edit the commands. In this file, they are not preceded by a quotation mark. For example:

/LSTOCKS,A	Will load the SuperCalc file STOCKS, all of it.
=A20	Jumps to cell A20.
"Title from MYEXEC	Adds a new title line.
/SSTOCKS,O,A,	Saves it as STOCKS, overwriting the previous file.

Quick Tip

Creating a command file in WordStar is easier because you do not have to precede each line with a quotation mark; you do not have to remove the border, and you can re-edit the command file whenever you like.

PART 4
CBASIC

Introduction

This part provides you with information about the CBASIC software that accompanies your Osborne Executive computer. If you've bought programs that require the use of CBASIC, you'll want to read "What CBASIC Is," "Creating CBASIC Programs," and "Compiling CBASIC Programs"; if you know how to program, you'll want to read all of this chapter. Otherwise, skip it for now.

What CBASIC Is

CBASIC is a version of the BASIC program language, which has been available for almost the entire history of microcomputers. The version of CBASIC you received is the current one, CBASIC2. Two earlier versions of the language were EBASIC and CBASIC.

CBASIC2 is a "pseudo-compiler." This means you create your program by using an editor (like WordStar) and then run your program through a portion of CBASIC, **CBAS2**, which translates your BASIC statements into a more compressed form. Once this "compiled code" is created, another portion of the CBASIC "interprets" these compressed instructions.

You need to understand two terms to use CBASIC:

"Source code" — your original BASIC instructions created using an editor, such as WordStar.
"Object code" — the compacted instructions CBASIC creates from your source code.

Creating CBASIC Programs

In the *Reference Guide,* you'll find a full list of the statements and functions that CBASIC understands, complete with a full discussion of what each is for and the exact syntax in which to use each.

If you're familiar with another version of BASIC, you'll probably find that CBASIC is similar. The most notable differences between CBASIC and other BASICs are the additional statements and functions CBASIC provides for use in CP/M.

CBASIC source code is written as a sequence of CBASIC statements and functions. The best method to create your source program is to use the **N** (non-document file) option in WordStar. You may give your file any legal name, but it must be the **.BAS** type. For example, a source program could have the file name:

PROGRAM.BAS

You should note a few things in creating your CBASIC source code. Since we're assuming you already know something about programming, we'll just briefly summarize the unique aspects of CBASIC.

- The CBASIC compiler ignores anything on a line to the right of a \ character. The \ character itself is an instruction to CBASIC to consider that the information on the following line should be part of this previous line. With the backslash character, you can create multiline statements. **DATA**, **DIM** and **END** statements must be on separate lines by themselves, however.

- CBASIC programs do not require line numbers except where they specifically reference a line, as in a **GOTO NUMBER** statement. Line numbers may be any valid number, and you can express them in exponential notation or with decimal points, although CBASIC treats the following numbers as different line numbers:

  100
  100.0
  1.0E02

- CBASIC variable names can have as many as 31 characters. If a variable name ends in a percent sign (%), CBASIC considers it an integer-only variable. Ending a variable name with a dollar sign (**$**) tells CBASIC that the name references a string variable.

- CBASIC has statements and functions not usually found in BASIC, or that you may not have encountered before. Most notably these are:

statements	**functions**
CALL	COMMAND$
CONSOLE	CONCHAR%
INITIALIZE	CONSTAT%
LPRINTER	FLOAT
PRINT USING	MATCH
SAVEMEM	SADD
WHILE—WEND	UCASE$

  Before attempting any significant CBASIC programming, you might want to look over the definitions for these statements and functions in the *Reference Guide.*

- Six compiler-directive statements in CBASIC expand your ability to create complex programs. These directives are also detailed in the *Reference Guide* and deserve your attention if you are doing complex CBASIC programming.

Building on your knowledge about how to program, the next chapter will supplement this brief introduction. Even with the complete *Reference Guide,* this manual will probably not suffice to teach you to use CBASIC to create programs.

We'll recapitulate what you have to know to program in CBASIC:

- You have to create CBASIC source code using an editor.
- CBASIC differs from other BASICs mostly in the additional statements and functions it provides.
- CBASIC has a few unique idiosyncrasies regarding line numbering, multiple-line statements, and variable names.
- CBASIC source code must reside in a **.BAS**-type file.

Compiling CBASIC Programs

To novice computer users who purchased a program that requires the use of CBASIC: the program you received is on a diskette containing instructions that constitute the program, but not the method by which to interpret them. CBASIC will be the interpreter for these programs.

Your CBASIC-based program may be in source-code format or in object-code format. The difference is critical. If you have source code, you must first "compile" the programs into the compacted form that CBASIC can interpret. If you have object code, the compacting is already done for you; the program is ready to use.

With source code, you can change the instructions should the need arise. Not so with object code. In fact, you can't even "list" these instructions to see what they are.

Let's assume for a moment that you have source code. You perform the actual "compilation," or compacting, by performing the following sequence:

1. Press RESET on the front of the Osborne Executive computer. Place your CBASIC/MBASIC diskette in the left-hand drive and press RETURN.
2. The Osborne logo will appear, and Microsoft BASIC will load. Once Microsoft BASIC identifies itself by displaying its name on the screen followed by an **Ok**, type:

   S Y S T E M RETURN

   You should see the CP/M prompt **A>**.
3. Place the diskette containing your CBASIC program in the right-hand drive. Type:

   C B A S 2 SPACE BAR B :
   f i l e n a m e RETURN

   Filename is the name of the program to compile. Do not include the file type when you enter the file's name. CBASIC will load its compiler portion into memory and begin converting the program in drive **B**. If you don't know the name of the program to be compiled, first type D I R SPACE BAR B : * . B A S RETURN to see the names of the CBASIC programs on drive **B**.

Note:

The instructions that accompany a CBASIC program coming from a software firm might tell you to type some special instructions—called "compiler directives"—after the file name. Be sure to do so or you may have an improperly functioning program.

4. You'll see a report on the screen at the end of compilation, and if it informs you that you made no compilation errors, your program is now ready to use. Make sure you compile all the **.BAS**-type files on the **B** drive—many complex programs actually consist of a series of "interlocking" short programs.

You should note two things in the above sequence. First, CBASIC displays the code it is compiling as it does so. CBASIC reports errors as it detects them, and summarizes them at the end of the process. If you want a "listing" of the program as it is compiled and have a printer attached to your Osborne Executive, substitute:

CBAS2 B:filename $F RETURN

for the instruction listed in step 3, above. ($F is one of those special compiler directives we mentioned.)

A special program called **XREF** is also on your CBASIC diskette. With this program you can create a list of all the variables the program uses and where it uses them. This ability is often useful in finding and eradicating any errors in a program. You invoke **XREF** like this:

X R E F SPACE BAR f i l e n a m e SPACE BAR
B : RETURN

XREF will create a new file with the same name, but its type is **.XRF**; it contains this special listing.

Using a CBASIC Program

To use your CBASIC programs, we advise you to create a special diskette, on which CBASIC and your programs reside, that you can load into the system just as you load your CP/M System, WordStar and SuperCalc diskettes. Let's learn how to do so.

Note:

You might want to refer back to this section after you've read the Microsoft BASIC chapter, as the process is valid for MBASIC as well as CBASIC.

1. Press the RESET button on the front of your Osborne Executive, put your system master diskette in drive **A** and press RETURN.
2. The Osborne logo will appear, and the **HELP** program will load. Press the ESC key to start CP/M.
3. Place a fresh, unused diskette into drive **B**.
4. Format the diskette in drive **B** using the **COPY** program as you learned earlier in this manual. If you wish to create several CBASIC program diskettes, go ahead and format several diskettes.
5. Return to CP/M and type the following:

   S Y S G E N RETURN

   A special program that copies CP/M "systems" will load into memory and identify itself. In response to its question about "source diskette," respond with an A. After a few moments, **SYSGEN** will ask you for a destination diskette; specify B, as that's where your CBASIC diskette-to-be resides, then RETURN. **SYSGEN** will access the **B** drive for a moment then ask you again about a destination. If you have more diskettes with which to perform this procedure, place them, one at a time, in drive **B** and continue to press the B key in response to **SYSGEN**'s prompting. When you're done, press RETURN. You've just written the instructions that constitute CP/M onto your new diskettes—a necessary action because CBASIC requires CP/M in order to operate.

6. Type P I P RETURN. You should see an asterisk on your screen after a few moments. Remove your CP/M diskette from the left-hand drive and replace it with your CBASIC/MBASIC diskette. Type:

   ```
   B : = A : C R U N 2 . C O M RETURN
   ```

   (For Microsoft BASIC, substitute **MBASIC** for **CRUN2** in the above example.)

   Both disk drives will be in use alternately for a few moments; then the asterisk will return. You've just copied the CBASIC interpreter onto your new diskette. You'll have to repeat this step for each diskette you create, but remember that to do so you have to tell CP/M that you've changed diskettes by typing C. This, unfortunately, causes **PIP** to stop running and return you to CP/M. A less-problematic alternative is to create one diskette and then use the **COPY** program to duplicate it.

7. You should still see the asterisk on your screen. Again place the CP/M system diskette in the left-hand drive.

Type:

```
B:=A:PIP.COM RETURN
```

then when the asterisk returns, type:

```
B:=A:XDIR.COM RETURN
```

then when the asterisk returns, type:

```
B:=A:COPY.COM RETURN
```

What you are doing is making copies of a few often used programs that will be convenient to have on your CBASIC program diskette.

8. You still have the asterisk prompt on your screen. Now transfer the CBASIC programs you compiled earlier. Place the diskette with the compiled programs on them into drive A and type:

   ```
   B:=A:*.INT RETURN
   ```

   (For Microsoft BASIC, substitute **.BAS** for the **.INT** in the above instruction.)

   This time the copying process will inform you as each program is copied.

9. Put the system diskette back into the left-hand drive and type ^C. CP/M should restart.

10. Your last step is to use the SETUP program to configure the necessary printer/special devices you intend to use with your CBASIC programs. Make sure you save these changes on the right diskette, the one in drive **B**.

One thing that might occur during one of the above steps is that you'll get a disk error message—such as **BDOS ERR ON**

A:—or a **DISK FULL** message. If this happens, you may have to omit one or more of the programs specified above when you copy files.

PART 5
Microsoft BASIC

Introduction

In this chapter you'll learn about Microsoft BASIC. You'll learn how to run programs that use Microsoft BASIC, and you'll find an introduction to the commands that make up Microsoft BASIC. This chapter won't teach you everything you need to know about programming in BASIC, but it will get you started. If you are serious about learning how to program, we suggest you browse through the book rack of your local computer store and select a book on BASIC programming with examples specific to your area of interest—such as business programming or scientific programming.

Starting Microsoft BASIC

In CP/M, starting Microsoft BASIC can be simple or complex, depending on the use you intend for it. Since you're just beginning, we'll show you the general manner in which to run Microsoft BASIC.

To start MBASIC, follow the sequence below:

1. Press the RESET button on the front of your Osborne Executive computer.
2. Insert the diskette that contains Microsoft BASIC and press RETURN as the instructions on the monitor tell you.
3. The Osborne logo will appear along with the words

```
        Loading Microsoft BASIC
```

After a few more moments you'll see a multiple-line message, which identifies Microsoft BASIC, and the letters **Ok.**

Use the above sequence if you just want to load BASIC to play around with it or to start working on a new program. If you already have a partially or completely finished program, you'll want to load it into your computer's memory along with Microsoft BASIC. You could follow the above sequence and then type:

L O A D " *program name* " RETURN

or

R U N " *program name* " RETURN

depending on whether you wanted Microsoft BASIC to bring the program into memory for examination/modification or to begin running it.

You may encounter a few errors if you attempt to load and/or run a program. Some of the more common ones are:

```
SYNTAX ERROR IN XX
```

This message means that Microsoft BASIC tried to run your program but came across an instruction it did not understand in line XX. Something in your program is not entered in a way Microsoft BASIC recognizes; examine the line in question while consulting the appendix that gives syntax examples for each statement.

```
STATEMENT NOT FOUND IN XX
```

This message indicates that Microsoft BASIC tried to run your program but could not find a line number the program referenced. You have a flaw in program control and should check your program while rereading the section on program control.

```
DIRECT STATEMENT IN FILE
```

You receive this message if Microsoft BASIC finds a statement that is not preceded by a line number when it tries to load your program. This error sometimes occurs if you create a program using the WordStar editor and accidentally forget to include a line number or leave extraneous material in the program; try reediting the program.

```
FILE NOT FOUND
```

If you get this message, it means that Microsoft BASIC tried to load your program, but couldn't find it. Check that you have the right diskette and that you specified the right program name.

Before we learn about the "building blocks" of the BASIC computer language, we need to ascertain that we're talking about the same thing when we say that you're learning how to "program" your computer.

Before you can build a computer program in BASIC, you must know what you are attempting to create.

A *computer* is a group of components that work together to perform tasks. A *program* is the instructions that tell the computer how to perform the task. To be more precise, let's define a program as:

> *the complete set of instructions necessary for a computer to fulfill a predefined function (or functions)*

Your goal in this chapter is to learn about the various instructions that comprise the BASIC language—specifically, Microsoft BASIC—then to learn how to group these instructions to perform a complete function or task. More technical information is packed into this chapter than any of the previous ones, so we'll forewarn you that you'll probably have to put more effort into its lessons. Furthermore, since so much detailed information is presented here, you will have to decide which examples to try. Don't lose sight of your ultimate goal: telling the computer how to perform a task or function.

Entering Programs

To understand how to enter a program using Microsoft BASIC, you must first understand a little about what makes up a program. In Microsoft BASIC, the basic component consists of what are known as "statements."

```
LINE# STATEMENT:STATEMENT
  ↑        ↑    ↑   ↑
  |        |    |   (optional additional statements)
  |        |    (optional colon separates statements
  |        |    on line)
  |        (instruction tells BASIC what to do)
(line number establishes order of instructions)
```

A statement is one complete instruction to BASIC to perform some action. The action can be assigning a value to a specific variable name, or something more complex such as formatting output according to a "mask" you design. When you press the carriage return, Microsoft BASIC stores each line for later use. A program consists of statements strung together in a particular order of execution.

Generally, each individual statement is on a line of its own. Each line must have a number associated with it. When a BASIC program "runs," lines execute in numeric order, unless you specify otherwise. So BASIC will perform instructions on a line numbered **10** before it performs the instructions on a line numbered **20**. For ease in changing a program after parts of it are written, you need not use consecutive numbers when you assign line numbers. Thus, the following example is a valid sequence of BASIC statements:

```
 10 STATEMENT (executes first)
 30 STATEMENT (executes second)
 32 STATEMENT (executes third)
 33 STATEMENT (executes fourth)
199 STATEMENT (executes fifth)
```

Any number between 0 and 65529, inclusive, can refer to a specific line in Microsoft BASIC. You must use actual digits to indicate the line number. Microsoft BASIC doesn't recognize typing **one** as meaning line number **one**. Note that a space must follow the line number to separate it from the instructions that follow.

It is also possible to have more than one instruction on a line. Each separate statement in such a case is separated by a colon, as in the example below.

```
LINE# STATEMENT1:STATEMENT2:STATEMENT3
```

In this example, when the line executes, **STATEMENT1** would execute first, followed by **STATEMENT2** and **STATEMENT3**, in that order. Some exceptions to this rule occur, and we'll state them in the discussion of commands that differ from the norm.

You can have any number of statements on a single line—each set off by a colon—up to a maximum of 255 characters. On the Osborne Executive, 255 characters slightly more than three complete display lines. The line number counts as part of your 255 characters.

In addition to the "program" mode described above, is an "immediate" mode of Microsoft BASIC. You can issue most Microsoft BASIC commands directly after you receive the **Ok** prompt. You can even issue multiple commands by setting off each with a colon, as described above. Your instructions will execute as soon as you press the **RETURN** key to indicate that you have entered a complete line. The 255-character rule also applies to the immediate mode, by the way.

You have available two ways to make Microsoft BASIC perform functions for you: 1) **immediate mode**—instructions execute after you enter a line of valid BASIC commands terminated with a **RETURN**, or 2) **program mode**—instructions will execute in the order of their associated line numbers when you later issue the **RUN** command.

For ease in entering programs, Microsoft BASIC has a built-in command that automatically generates line numbers for you as you're programming; this command is called **AUTO**. You issue it in the immediate mode, thus:

AUTO x,y RETURN

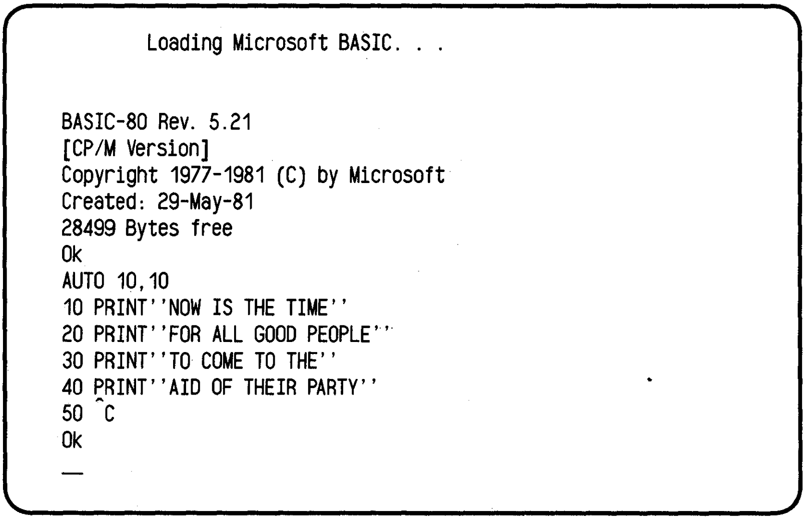
where **x** is the line number with which you wish Microsoft to begin your entry sequence, and **y** is the increment to use for successive line numbers. Typing:

A U T O SPACE BAR 1 0 , 1 0 RETURN

tells Microsoft BASIC to begin with line number 10 and to increment subsequent line numbers by **10** (i.e., the second line number will be **20**, the third, **30**, and so on).

When you use the **AUTO** command, Microsoft BASIC displays the current line number and waits for you to enter the instructions for that line. When you complete the line by pressing **RETURN**, BASIC will provide the next line number automatically and wait for you to enter the instructions for that line. This process continues until you type a control-C as the first character on a line, at which time Microsoft BASIC drops out of the auto-line-numbering mode and returns to the immediate mode. Since you did not enter any valid instructions on the last line number displayed, your program won't include it.

To help you understand the process described above, here's a complete example of **AUTO** being used to program:



```
        Loading Microsoft BASIC. . .


BASIC-80 Rev. 5.21
[CP/M Version]
Copyright 1977-1981 (C) by Microsoft
Created: 29-May-81
28499 Bytes free
Ok
AUTO 10,10
10 PRINT''NOW IS THE TIME''
20 PRINT''FOR ALL GOOD PEOPLE''
30 PRINT''TO COME TO THE''
40 PRINT''AID OF THEIR PARTY''
50 ^C
Ok
_
```

The program you entered would look like this:

10 PRINT"NOW IS THE TIME"
20 PRINT"FOR ALL GOOD PEOPLE"
30 PRINT"TO COME TO THE"
40 PRINT"AID OF THEIR PARTY."

Line 50 does not exist; you didn't place an instruction on that line—the control-C told BASIC you wanted to quit.

Before we leave the **AUTO** command, you need to learn that if you omit either of the numbers the command expects, Microsoft BASIC will assume you want the default value, 10, meaning that:

```
AUTO  10 is the same as AUTO 10,10
AUTO     is the same as AUTO 10,10
AUTO ,10 is the same as AUTO 10,10
```

Here the topic of how to number lines when you're programming arises. The default values that Microsoft BASIC assumes—line numbers starting at **10** and progressing in intervals of **10**—are a good starting point because you can create any line numbers you wish, and the program will execute in the order of those numbers. What happens if you enter the following program?

```
1 PRINT"NOW IS THE TIME"
2 PRINT"TO COME TO THE"
3 PRINT"AID OF THEIR PARTY."
```

You've accidentally left **FOR ALL GOOD PEOPLE** out of the quote. Since you've used line-number intervals of one, in order to tell BASIC to print the missing information you'll have to create a line number **1.5**. Since Microsoft BASIC only recognizes whole numbers to identify lines, you'll now have to reenter lines 2 and 3 in the above example and assign them higher line numbers to replace the missing part of the quote.

> You've just learned the first of what we might call **Murphy's Laws of BASIC Programming**: If you need to insert two lines between existing ones, you'll find you have room for only one.

Wise programmers choose their line numbers carefully. You should follow these rules for choosing line numbers:

1. Each major section of a program starts on a line number that is divisible by 500.

2. Each major "subroutine" within a program starts on a line number that is divisible by 500 and greater than 10000.
3. Otherwise, use line numbers with intervals of 10.

Follow the above rules and you can add extra lines, sections, or subroutines at a later date without having to renumber any existing line.

While we're on the subject of line numbers, you might wonder how you get rid of an existing line. You have two choices:

1. Type the line number you wish to eliminate, followed by a carriage return (i.e., an empty line).
2. Use the **DELETE** command described in a later section of this manual.

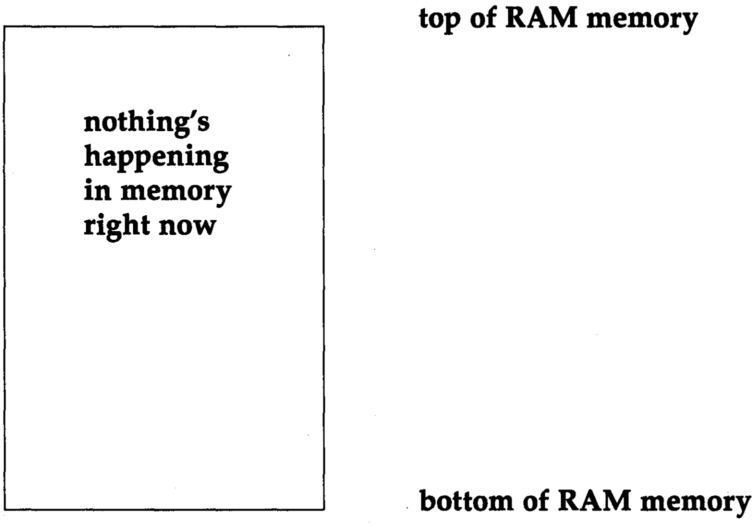
Okay, you should be all "lined up," ready to learn about variables and begin programming.

Introduction to Variables

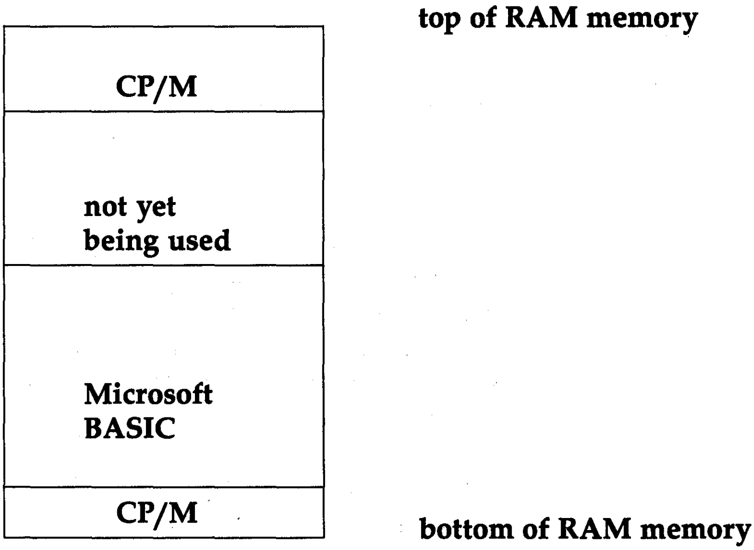
Your computer has a lot of "memory," but how do you make use of that storage capability.

Actually, when you load CP/M and Microsoft BASIC into your computer, you already begin using some of the RAM (random-access memory) your system contains. Before we show you what variables are and how BASIC maintains them, we want to make sure you know what's happening inside your computer.

If you could have looked inside the Osborne Executive's memory when you first turned it ON, you would have found that nothing had yet been stored there:



If you could have looked inside your memory after you loaded and executed CP/M and Microsoft BASIC, you would have found this:



As you can see, CP/M takes up a little memory at the bottom (256 bytes) and some more room at the top—usually between 4K and 8K, depending on the version of CP/M. Microsoft BASIC, which has loaded toward the bottom of memory, uses approximately 24K of memory area. The portion of memory labeled **not yet being used** is where your program and variables (information) will be stored.

Generally, you don't need to know exactly where each piece of your program is stored. We will point out, however, that Microsoft BASIC uses the space immediately following the program and works its way up as it needs to use more memory.

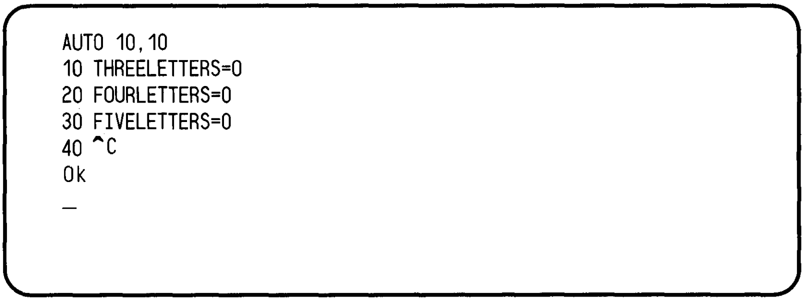
A variable is a temporary storage area reserved for information used within a BASIC program. As you program, you assign arbitrary names to each variable, and it is by this variable name that you can tell BASIC what to do with the information.

We'll show the concept of variables with an example. Suppose that you were trying to count the number of three-, four-, and five-letter words in this paragraph. If you were counting in your head, without the benefit of a pen and paper to make "tally marks," you'd probably say the following to yourself:

> "That's no three letter, one four letter, no five letter; that's no three letter, two four letter, no five letter; that's one three letter, two four letter, no five letter."

Consider this example carefully. You're using placement and identification to keep track of each changing piece of information. When you're programming, BASIC keeps track of the placement, while you use the identification to tell your program when to update a piece of information.

Let's look at how you do this in Microsoft BASIC:



```
AUTO 10,10
10 THREELETTERS=0
20 FOURLETTERS=0
30 FIVELETTERS=0
40 ^C
Ok
_
```

To start, each variable has a name and is set equal to zero. The first time it encounters a variable name, Microsoft BASIC assumes that the value is zero, unless you inform it otherwise, but it is good programming practice to explicitly declare a value for every variable at the beginning of your program. That way you'll be sure that the values used are the ones you intended.

Note that our choice of names is purely arbitrary. Within certain constraints, Microsoft BASIC allows you to assign names that are any combination of letters up 40 characters long. Actually, you could have longer names, but BASIC would only use the first 40 characters.

At this point, you only need to know a few rules about choosing names:

1. All names must begin with a letter.
2. Make your names between 1 and 40 characters in length.
3. Do not start a name with the letters **FN**.
4. Do not use special characters (**@**, **#**, **$**, **%**, **^**, **&**, etc.); use only uppercase letters and numbers.
5. Variable names cannot be any word already in use as an MBASIC command.

Now, back to our example.

We introduced you to the most used BASIC command without your realizing it. Remember the statement **10 THREELETTERS = 0**? Did you wonder what the **= 0** part of it meant?

All three statements in our example are called **LET** statements. The general form of a **LET** statement is:

```
LINE# LET X = Y
```

As in our example, you need not type **LET**; the equal sign in the statement implies it. The above means: 1) calculate or retrieve variable **Y**; 2) move its value into the storage location assigned to **X**.

Acquire the habit of saying to yourself, "New equals old," because we're going to confuse the picture a bit with a new statement for our example:

```
50 THREELETTERS = THREELETTERS + 1
```

In the above statement, using the first rule we just stated, BASIC will retrieve the value of **THREELETTERS**, add one, and then store the result in **THREELETTERS** (following the second rule).

LET statements can be simple, like the one above, or they can be quite complex, like the one below:

```
1010 RESULT = ((XPOS*YPOS)-(42/ZPOS))/2
```

The above calculation uses algebraic notation. For the most part, Microsoft BASIC follows algebraic notation conventions exactly. Below are some of the rules of evaluating algebraic expressions:

1. Do all multiplication and division before addition and subtraction operations.

   $$\mathbf{A + B +} \underset{\substack{\uparrow \\ \text{calculated first}}}{\underline{\mathbf{C * D}}}$$

   We'll elaborate on "precedence" in a few paragraphs, when we introduce some other arithmetic operations.

2. Calculate the expressions in the innermost set of parentheses first.

   $$\mathbf{(A + (B * C +} \underline{\mathbf{(D}} \underset{\substack{\uparrow \\ \text{calculated first}}}{\mathbf{/}} \underline{\mathbf{E)}}\mathbf{))}$$

3. Work from left to right if none of the above rules apply.

   $$\underset{\substack{\uparrow \\ \text{calculated first}}}{\underline{\mathbf{A + B}}} \mathbf{+ C + D + E}$$

Using the above rules, if you were calculating the expression

$$\mathbf{(24 * 5) / 2 - 5 + 10}$$

you'd perform the following steps:

1. multiply 24 times 5 (equals 120)
2. divide by 2 (equals 60)
3. subtract 5 (equals 55)
4. add 10 (equals 65)

In Microsoft BASIC, the statement that represents the above calculation looks like this:

```
200 RESULT = (24*5)/2-5+10
             or
200 RESULT = 24*5/2-5+10
```

While we're concentrating on things mathematical, let's make sure you know the basic arithmetic functions a **LET** statement can use, in order of their precedence during calculation:

^ represents exponentiation: **2^2** means "two raised to the power of two"; exponents are calculated before multiplication and division are performed, but otherwise follow the rules stated above.

* represents multiplication: **2*2** means "two times two."

/ represents division: **2/2** signifies "two divided by two."

\ represents integer division: **2\2** means "the integer result of two divided by two," or that any remainder is forgotten after the division is completed. Integer division takes precedence over addition and subtraction, but not over multiplication and division.

MOD represents modulus division: **2 MOD 2** denotes "the integer remainder left after two is divided by two"; modulus division is performed after integer division, but before addition and subtraction.

+ represents addition: **2+2** means "two plus two."

- represents subtraction: **2-2** signifies "two minus two."

One other "operator" you should know about is called "negation." Negation looks like subtraction in that it uses a hyphen (-) to indicate its presence. You can tell whether the ⊟

implies subtraction or negation by the context in which it is used. Look at the example below.

X - Y	subtraction
X * (-Y)	negation (parentheses must separate two consecutive operators . . . one hint that negation is implied)

Before you leave **LET** statements, note just a few more things about them. First, you can freely intermix variables with explicitly stated values, as in this example:

100 RESULT = 45 * VARIABLE2

You must, however, place a variable on the left side of the equal sign. In fact, only one element can go on the left side of the equations; this is another requirement of the LET statement: you can only assign values to one variable at a time. Thus, none of the following are valid statements:

10 45 = RESULT * SECONDRESULT	◀doesn't have a variable to the left of the equal sign
10 RESULT + SECONDRESULT = 45	◀you've assigned two variables on the left of the equal sign

Do You Know the Variable Type?

So far, we've used only examples in which numbers or values are stored in variables. Actually, you can specify four types of variables in Microsoft BASIC.

1. **INTEGER** variables can store only values between −32768 and 32767. Integer numbers cannot have embedded decimal points.

2. **SINGLE-PRECISION** variables can store any positive or negative real numbers, including those with fractional values indicated by a decimal point (i.e., 3.1456). Single-precision variables are so called because they store only the seven most significant numbers and print with only the six most significant. If you overstep these bounds, the result will be incorrect rounding or truncation of information, depending on the operation you are performing.

   45 / 7 = 6.4285714 with 9-digit precision

   but the result is only

   6.42857 with 6-digit precision

   The rounding in the above example is correct—at least if you're concerned only with the first six digits of accuracy—but what would happen if the result stored were 6.4285771? Precision errors have a tendency to mushroom. Although no error may exist after one operation, repeated operations can exaggerate any imprecisions.

3. **DOUBLE-PRECISION** variables not only store any positive or negative real number, but they can also do so with 16 digits of accuracy.

4. **STRING** variables are a special type. Even though they can store certain types of values, they almost always store "characters" of information. You assign information to be stored in a string variable by using quotes around the characters you wish to store:

   100 STRINGVARIABLE$ = "ABCDEF"

The above example places **ABCDEF** in storage with the name STRINGVARIABLE$ assigned to it.

You can do more than make assignments with string variables. In a later lesson, we'll show you how to add, subtract, compare, and parse, among other functions, with string variables.

You may wonder how BASIC can know which type of variable you're referencing when you use a name. Actually, if you do not explicitly state how you want BASIC to store information, it will assume that your variables are single-precision.

To tell Microsoft BASIC the type of variable you desire, append a single character to the end of the variable name. The character informs BASIC of your intentions.

%	integer variable
!	single-precision variable
#	double-precision variable
$	string variable

METOO%	
METWO%	all integer variables
METO%	
YOUANDME	Note: no symbol means single-precision, also
YOUANDME!	all single-precision variables
UANDME!	
HE#	
SHE#	all double-precision variables
IT#	
MONEY$	
DOLLAR$	all string variables
MULLAH$	

Once you define a variable using an extra character, you must continue to employ that character every time you use the variable in your program. **HE#** and **HE**, for example, are interpreted as different variables.

You might sometimes notice that as you are typing a program, Microsoft BASIC supplies an exclamation point or number sign that you didn't specify. For instance, typing the following statement:

```
5440 ANDHIKE = 654578
```

will appear in any program listing as:

```
5440 ANDHIKE = 654578!
```

If you had declared **ANDHIKE** a single-precision variable, the exclamation point would not have appeared after the specified number.

We don't want to bog you down in arithmetic details this early in the chapter, so we'll save supplemental details for later. Be forewarned, however, that you haven't learned everything there is to know on the subject of Microsoft BASIC's variable notation.

Print It

So far you only know how to number lines and assign values to variables. Now we'll show you a way to get information out of the computer.

Enter each line of the following program lines, pressing RETURN after each:

```
10 VARIABLE2 = 10
20 RESULT = 45 * VARIABLE2
```

It would be nice to learn the value of **RESULT**, and it's simple:

you use a **PRINT** statement. Type:

```
30 PRINT RESULT
```

followed by RETURN. If you now run the program—type ^C and R U N, then press RETURN—you'll see **450** appear on your screen. Try this new statement. Type:

```
A U T O SPACE BAR 1 0
```

and press RETURN, then type:

```
VARIABLE2 = 10
RESULT = 45 * VARIABLE2
PRINT RESULT
```

follow each line with RETURN and on line 40 type ^C followed by R U N. Microsoft BASIC will display the value of **RESULT 450** and follow that with another **Ok**.

PRINT statements can be as complex as **LET** statements. Instead of adding line 30 in the example, we could have made our program:

```
10 VARIABLE2 = 10
20 PRINT 45 * VARIABLE2
```

Don't rush to any conclusions about this last point, though. Sometimes it pays to shorten your program by combining **LET** and **PRINT** statements—as we have just done—sometimes it doesn't. Generally, if a value is going to be calculated only once, it is okay to calculate it on a **PRINT** statement. In the interest of clarity (does someone else have to be able to understand your programs?), however, you should avoid the combining practice.

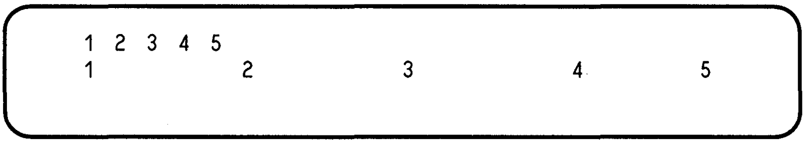
Two other aspects of the **PRINT** statement are important. First, if you wish to send something to a printer instead of the

Osborne Executive screen, substitute the word **LPRINT** (Line PRINTer) for **PRINT**. For this procedure to work, you must have a printer attached to your Osborne Executive and have used the **SETUP** program to specify what kind of printer you're using. In addition, the printer must be turned ON and ready to print; don't scoff, many so-called "problems" with printers arise because no one bothered to turn the printer ON!

Another significant aspect of **PRINT** statements is that you can use one **PRINT** command to print multiple items. Each item to print is set off by a semicolon or a comma, depending on whether you want the items to print with 14 or no spaces between them. Enter the following:

```
10 PRINT 1;2;3;4;5
20 PRINT 1,2,3,4,5
```

On line 30, type ^C to get in the command mode, then type R U N, and press RETURN. You should see:



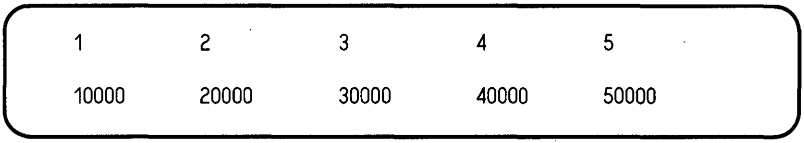
```
 1  2  3  4  5
 1             2             3             4             5
```

Hold on. Didn't we just say no spaces would appear between the values if you used the semicolon? True, but Microsoft BASIC is leaving a space in the printout for the "sign" (positive/negative) of the number. Since the numbers are positive, and BASIC doesn't show a + sign, you should assume from this that BASIC exhibits only negative signs unless you explicitly tell it to display the positive sign for a value. In addition, when you print numbers, each number is always followed by at least one space, even if you use the semicolon.

Let's look at the use of the comma in a **PRINT** statement a little more carefully, too. Enter the following:

```
10 PRINT 00001,00002,00003,00004,00005
20 PRINT 10000,20000,30000,40000,50000
```

On line 30, type ^C, type R U N and press RETURN. You'll see:



```
1             2             3             4             5
10000         20000         30000         40000         50000
```

If you count carefully, you'll find that each new item starts 14 spaces after the preceding one. Regardless of the length of the item, the numbers print at the beginning of 14-character columns.

Some final tidbits about the **PRINT** statement: for the word **PRINT**, you can substitute a question mark. This saves typing four additional characters. Also, Microsoft BASIC can understand **PRINT**, even if a line number doesn't precede it. Enter ? 4 5 * 1 0 followed by RETURN. You should see the result just as before. This last feature is handy when you're trying to calculate a value but don't want to write a program to figure it out. Think of this feature as an expensive calculator.

Remarks

Sometimes you merely want to insert into a program a comment—something that suggests what is happening but that does not appear to BASIC as an instruction to do something. For such comment insertions, use the **REM** (remark) statement:

```
10 REM This is the start of the program
```

It is wise to use plenty of remarks in your program when you are still building it—you can take them out later if the program starts to become large or cumbersome.

You can add remarks to lines that contain instructions:

```
10 PRINT HRS;:PRINT SECS:' Tell user how much time
has expired
```

What?! We pulled three tricks on you in the above example: 1) we substituted a ' for **REM**, 2) we put three instructions on one line (two **PRINT**s and one **REM**; and 3) we made the hours (**HRS**) and seconds (**SECS**) print together because we ended the first PRINT statement with a semicolon.

The first trick is the same as replacing **PRINT** with a question mark, and Microsoft BASIC recognizes it as a valid substitution.

To perform the second trick, set off each statement with a colon. Thus, our example is the equivalent of:

```
10 PRINT HRS;
11 PRINT SECS
12 REM Tell user how much time has expired
```

The third trick should make sense to you. Since the semicolon—when used between items to be printed—inserts no spaces between the items, the semicolon will perform the same function, even if the items to print are in different statements; the same is true of the comma.

If you're concerned that BASIC is complex and you have too many things to remember, don't worry. We've already introduced about ten percent of Microsoft BASIC to you, and this fraction of the language will probably account for over half of your BASIC programming.

To round out this lesson, we'll teach you some other ways to manipulate statements. Type [N][E][W][RETURN], then [A][U][T][O][SPACE BAR][1][0][RETURN], and enter the following program:

```
PRINT"The clothes are a great buy."
PRINT"They only cost:"
DISCOUNT=305
PRICE=1700
PRINT PRICE-DISCOUNT
END
```

When entered, type ^C and R U N, then press RETURN.

Already you've encountered two new statements:

NEW erases from memory any previously existing program. Use this with caution!

END tells Microsoft BASIC that it has come to the end of a program. For reasons that will become apparent later, this should be the last statement every program you write executes.

Take a close look at what the program printed on the screen. Are you happy with the results? We aren't: there's no dollar sign in front of the **1395**, and the price really should be on the line where **They only cost:** is. Is there any way to make the program function the way you want without having to type it all again?

To this question there is a reasonable answer. Type L I S T followed by RETURN. Microsoft BASIC should redisplay your program. Type L I S T SPACE BAR 1 0 followed by RETURN. Microsoft BASIC should display line 10 of your program. Type L I S T SPACE BAR 1 0 - 3 0 followed by RETURN. Microsoft BASIC should display lines 10–30 of your program.

Note:

To make the listing of your program, go to your printer instead of to the screen, issue **LLIST** instead of **LIST**. This construct is similar to the **LPRINT** statement described earlier.

Okay, so now you can relist your program to see what Microsoft BASIC thinks is in it. How do you change it?

You can enter in new lines as necessary, so let's change the way the end of the program functions. Enter the following, remembering to follow each line with a [RETURN]:

```
45 REALCOST = PRICE - DISCOUNT
50 PRINT REALCOST
```

List your program again to make sure that the changes are made correctly. Run your program again to ascertain that it works the same way. Does it? Good, now proceed to make some "cosmetic" changes. Type:

[E][D][I][T][SPACE BAR][2][0][RETURN]

You should see the line number (20) and nothing else but your cursor. Microsoft BASIC has retrieved line 20 from memory and is now waiting for you to enter one of the "editing" commands it recognizes:

SPACE: moves cursor forward one character and shows previous character.

LEFT ARROW: moves cursor backward one character but does not erase any characters.

I: tells Microsoft BASIC to begin "inserting" characters at the current cursor position. All characters to the right of the cursor will "push ahead" to make room for the new characters you enter.

ESC: ends the insertion of characters and resumes normal editing.

X: places cursor at the end of the current line and allows you to add to that line.

D: deletes the character at the current cursor position. So that you can tell deleted characters from retained ones before you execute the command, BASIC displays deleted characters in backslashes.

H: deletes all characters from the current cursor position to the end of the line.

Schar: typing **S** followed by another character tells Microsoft BASIC to go to the next occurrence of the character; the cursor positions just before the next occurrence.

Kchar: deletes all characters from the current cursor position to the next occurrence of the character specified.

C: replaces the next character in the statement with the next character you type.

RETURN or E: ends editing of the current line with all changes being used.

Q: quits editing of the current line without implementing the changes you may have made to the line.

L: lists the remainder of the line and repositions the cursor at the beginning of the line.

A: restarts the editing of a line; useful when you've made some changes you want to "take back."

Don't worry too much about memorizing the above commands. Eventually they'll be normal to you, as each single-letter command implies the editing function (**C**, Change; **D**, Delete; **L**, List). If you've learned WordStar, the actual functions performed should seem quite natural to you.

With the above commands in mind, return to that program with the pricing problem. We want to add a semicolon to the end of line 20 so that the price prints to the right of the **They only cost:** element. Issue the following sequence to do so:

E D I T SPACE BAR 2 0
X
;
RETURN

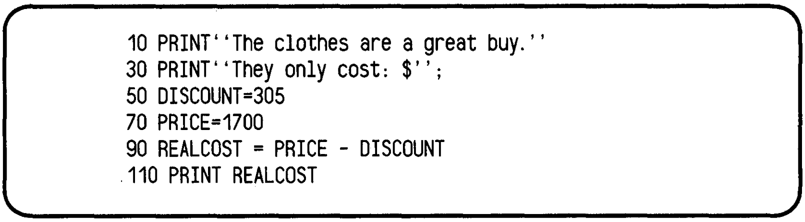
Now rerun the program and make sure it works as you want it to. It should look better, but still not correct. Where's the dollar sign? Change that line again; the sequence of commands you type this time is:

E D I T SPACE BAR 2 0
S " S "
I $ ESC RETURN

You just inserted a space and a dollar sign at the end of the information to print. Run the program one more time to make sure that it works.

Now get rid of a line by using the **DELETE** statement. Type D E L E T E SPACE BAR 6 0. Now list the program; line 60 should be missing. **DELETE** works as does **LIST**, although it always requires a single line number or a series of line numbers.

The subject of line numbers evokes Murphy's Laws of Programming, one of which states that if you need to insert two lines between two others, you'll only have room for one. The people at Microsoft must have heard of Murphy, because they've provided a **RENUM** command, which renumbers lines. Type R E N U M SPACE BAR 1 0 , 1 0 , 2 0, and press RETURN. Now list your program. You should see:



```
10 PRINT''The clothes are a great buy.''
30 PRINT''They only cost: $'';
50 DISCOUNT=305
70 PRICE=1700
90 REALCOST = PRICE - DISCOUNT
110 PRINT REALCOST
```

Notice that line 45 has been renumbered with the same multiple (20) as the rest of the lines. The components of the **RENUM** command are:

RENUM #1,#2,#3

where #1 is the first number to use in the new sequence;
#2 is the first current line number to change;
#3 is the increment to use for numbering lines.

You can omit any of the three, but be sure to use a comma to show its place if you are omitting only one or two of the components.

Despite all the abilities of the Osborne Executive, there are a few things you cannot do:

- use renumbering to rearrange the order in which statements occur, and
- create numbers greater than 65529.

You'll later learn about other statements, such as **GOTO** and

GOSUB, that at reference line numbers. If the line numbers these other statements reference exist, **RENUM** automatically changes the references.

Some Cleanup Instructions

You've now completed two of the 13 topics we want to present concerning Microsoft BASIC. So you have something to "play" with for the remaining lessons, make sure you know how to save your program for later use.

We're assuming that you have your Microsoft BASIC diskette in drive A. Type:

```
SAVE"PROGRAM1"
```

and press RETURN. Note that your program's name (**PROGRAM1**) must be enclosed in quotes. To save the program on drive B, type:

```
SAVE"B:PROGRAM1"
```

and press RETURN.

Control Structures

You know about variables, line manipulations, printing of information, arithmetic operations, and the **LET** statement. These attributes comprise about half of most simple programming chores in BASIC.

What you don't yet know is how to make BASIC execute only sections of a program. We'll call this "forced execution" "control structures" because you are creating a program structure that controls which statements to execute and which not to execute.

You want to retrieve that program you've been working on. Type:

LOAD"PROGRAM1"

and press RETURN. Next type LIST followed by RETURN to verify that the program loaded correctly. This exercise is more for your benefit than anything else, as your Osborne Executive usually reports any problems it encounters in loading a program file. To give yourself some space to work, renumber the program again:

RENUM SPACE BAR 100,10,10 RETURN

The result of this command is that your program now starts with line number 100, and each succeeding line number is in increments of 10.

Add the following to your program:

```
10 REM This is the new start of the new program
20 GOTO 100
30 END
```

Notice the **END** statement in line 30. If Microsoft BASIC sees this statement, the program stops executing. Notice also the **GOTO 100** statement in line 20. Since Microsoft BASIC executes a program by looking at line numbers in order, it will see **GOTO 100** before it sees **END**. The **GOTO** statement forces execution to take place at the line number indicated, so Microsoft BASIC never reaches line 30. Verify this by running the program; it should still work as before.

Just for fun, enter 160 GOTO 10 and press RETURN. Now run the program. The program goes around and around, printing the same message on the screen. Your program is now in an "infinite loop," so named because there is no possibility that the program will come to an ending place—it merely keeps on going back to the beginning and repeating the same thing over and over again. Depending on what you're trying to do, infinite looping can be both useful and frustrating.

Want to stop the looping? Press ^C to tell Microsoft BASIC to interrupt what it is doing and return control back to you. You will see a message such as **Break in 100**, to indicate what line number Microsoft was executing when you stopped it. Now issue: C O N T (for continue), and press RETURN. The message resumes where it left off. Press ^C to stop again.

Although that exercise was interesting, all you've learned so far is to command Microsoft BASIC to do something, period: "forced execution."

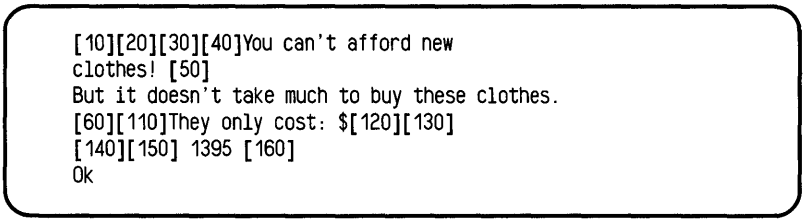
Another type of execution is called "conditional": execution resumes at a location dependent upon what "condition" Microsoft BASIC encounters. Let's examine this idea more carefully.

"If I were rich, I'd buy new clothes": that statement exhibits conditional execution. **If** you are rich, you'll buy new clothes; if you're not rich, the implication is that you won't. Let's add this construct to our program:

```
160 END:'CANCEL THE CONTINUOUS RUNNING
20 RICH=1
30 IF RICH=1 THEN GOTO 100
40 PRINT"You can't afford new clothes!"
50 PRINT"But it doesn't take much to buy these clothes."
60 GOTO 110
```

Run your program now. It still does the same thing, because at line 30, Microsoft BASIC "tests" to see if **RICH=1** or not. If that statement is true, then execution proceeds at line 100; if not, execution proceeds with the next line, 40. Now change line 20 to read **RICH=0** instead. What happens when you run the program now? It executes lines 40-60, skips line 100, and executes lines 110-160.

If you don't believe what you just read about which line numbers were executed and which were not, we'll teach you a trick you can use to verify which lines BASIC executes. Type T R O N (for trace on) and press RETURN; then issue R U N and press RETURN. You should see:



```
[10][20][30][40]You can't afford new
clothes! [50]
But it doesn't take much to buy these clothes.
[60][110]They only cost: $[120][130]
[140][150] 1395 [160]
Ok
```

Each of the bracketed numbers is a line number that executed. Although the example above clutters up the display, it allows you to verify exactly what Microsoft BASIC is doing at any point in the program. You might not need this facility now, but when your program becomes difficult to follow, **TRON** is a handy tool. To turn this feature off, type T R O F F (for trace off) followed by a RETURN.

The complexity of the **IF** test is up to you. The tests you create work about the same way as does the **LET** statement, introduced earlier. For example, the following is a valid **IF** test:

IF ((RICH=1) AND (POOR<>1)) THEN 100

Notice the use of parentheses to force BASIC to evaluate certain portions of the statement before others. Also, any of the "logical operators" that follow can be used:

<>	not equal to
=	equal to
<=	less than or equal to
=>	greater than or equal to

>	greater than
<	less than
AND	both subequations must be true
OR	one of the subequations must be true

The last two possibilities may be past your understanding at this point, but don't worry—we'll return to them at an appropriate time.

We want to show you one more trick concerning the **IF** test. Enter the following replacements for lines 20 and 30:

```
20 RICH=1
30 IF RICH THEN 100
```

We're using a lot of shorthand here. First, the test is missing! The reason this trick works is because Microsoft BASIC evaluates the entire expression between the **IF** and the **THEN**; if the expression is equal to zero, then the expression is false; if the expression is not equal to zero, the expression is true. To BASIC, a value of zero is always "false," while any other value is "true." Another trick is that we've left off the **GOTO** in line 30. In this example the context implies **GOTO**.

One other attribute of **IF** before we move on: you can insert any valid expression between **IF** and **THEN**, and any valid BASIC statement can follow **THEN**. Thus,

```
30 IF RICH<>1 THEN END
```

is a valid statement.

A variant of the **GOTO** statement you learned earlier is the **GOSUB** statement; **GOSUB** stands for "go to subroutine and return." **GOSUB** means almost the same thing as **GOTO**, but a **GOSUB** always comes back to the statement that follows it. You might compare the **GOSUB** statement to a **GOTO** boomerang—it always comes back. Let's kill two more birds while learning one new command:

```
5 GOSUB 1000
1000 PRINT CHR$(26)
1010 RETURN
```

Now run your program. The screen should clear, then your message should appear. Line 1000 clears the screen; for the time being, never mind how. Line 5 tells BASIC to execute statements that begin with line 1000, but to remember to return when it has finished the "subroutine" beginning at line 1000. Line 1010 contains the command that tells BASIC that your subroutine is complete and it's time to return to the main program's next statement.

In other words, **GOSUB** directs BASIC to execute a new section of the program—so far, just as **GOTO** does—but when BASIC encounters the magic word **RETURN, GOSUB** instructs BASIC to come back and begin executing the instructions that immediately follow the original **GOSUB**. Sound confusing? It shouldn't, especially since you already know the statements to use to check what's happening: turn the trace function on, run your program, and list it.

Note:

*You won't see the first two statements that execute because the screen clears after they've been passed; sorry about that, but let's move on to another control structure: **FOR/NEXT** loops.*

So far, the control structures we've introduced are simply "go from here to there (and maybe return)" types. The **FOR/NEXT** construct is an example of a structure called "looping."

A loop is a section of code that repeats. Remember the "endless loop" you created earlier in this lesson using the

GOTO statement. That program performed instructions, then looped back to the first one and started over again. We'll do the same thing with **FOR/NEXT**, except we'll tell the computer how many times to loop.

```
10 FOR LOOP=1 TO 10
20     PRINT"The clothes are a great buy."
30     PRINT"They only cost: $";
50     DISCOUNT=305
70     PRICE=1700
90     REALCOST = PRICE - DISCOUNT
100    PRINT REALCOST
110 NEXT LOOP
```

This new version of our program has some additional qualities. First, there is a variable named **LOOP** involved in the **FOR** statement; there could be others to the right of the equal sign, as well. Second, the program instructions ("code") that will execute each time the loop executes are contained between the **FOR** and the **NEXT** statements. Also, we have indented (using the **TAB** key) the instructions within the loop, so that the actual loop becomes more visible to the casual observer. This last point is not frivolous. In complex programs, loops often become buried in instructions, where they are difficult to find and check for proper functioning.

Several components make up the **FOR** and **NEXT** statements; we want to be sure you understand each part:

FOR variablename = startnumber TO endnumber

NEXT variablename

variablename is a name you give the loop;
startnumber is a starting value for the loop count; and
endnumber is the ending value for the loop count.

Every **FOR/NEXT** loop must have a variable name associated with it. During each iteration of the loop, the current loop number will be the value of that variable, and you can use that variable in calculations, if you want. Type:

```
FOR LOOP=1 TO 10
    VALUE = 2 * LOOP
    PRINT VALUE
NEXT LOOP
```

You have to specify the starting and ending count, although another variable can set these values.

```
FOR LOOP=START TO LAST
    VALUE = MULTIPLIER * LOOP
    PRINT VALUE
NEXT LOOP
```

Make sure, however, that you initialize (set the starting values of) the variables used for the starting and ending count before you get to the loop. A loop will not execute if its ending count is lower than its starting count. A special circumstance arises when the two counts are the same: the loop executes once. Because the starting count is not greater than the ending count, execution of the loop proceeds. When execution reaches the **NEXT** statement, the loop-counter variable has equalled the end value; therefore execution stops.

You may "nest" loops within loops. Each loop must be wholly contained within another—they cannot "overlap." Also, each loop must have its own name. Try entering and executing the following program; but before you do, see if you can guess what will print at each step of execution:

```
10  FOR LOOP1=1 TO 10
20      FOR LOOP2=1 TO 10
30          FOR LOOP3=1 TO 10
40              PRINT LOOP1,LOOP2,LOOP3
50              FOR DELAY=1 TO 100
60              NEXT DELAY
70          NEXT LOOP3
80      NEXT LOOP2
90  NEXT LOOP1
100 STOP
```

We sneaked a new statement into that last program example, by the way: **STOP**. **STOP** performs the same function as does typing a control-C while a program is running: it stops the program temporarily. To continue the program (if there's more, which there isn't in this example), simply type **CONT** and press **RETURN**, and execution picks up with the statement following the **STOP** statement.

STOP is handy to use when you're trying to figure out why a program doesn't work correctly. Insert a few **STOPs** into your program, and, when execution stops, check the values of the variables in your program by using the **PRINT** command in the immediate mode (without line numbers). Then enter **CONT** to resume where you left off.

Getting Around on the Diskette

You already know how to save and load BASIC programs. Before continuing with out programming examples, we want to teach you a few new tricks to do with the SAVE and LOAD statements, and briefly introduce some other commands for manipulating files on a diskette.

In CP/M, to get a directory of the files on a diskette, type **DIR** and press **RETURN**. Sometimes you use the asterisk and question mark to search for certain types of names (as in **DIR *.BAS**). In Microsoft BASIC, the **FILES** command does the same thing. The only difference between **FILES** and **DIR** is that you must specify any additional factors enclosed within quotation marks. The chart below shows equivalent commands in BASIC and CP/M.

BASIC	**CP/M**
FILES <cr>	DIR <cr>
FILES "A:*:*" <cr>	DIR A: <cr>
FILES "B:*.BAS" <cr>	DIR B:*.BAS <cr>

<cr> stands for **RETURN.**

You can use the **FILES** command within a BASIC program.

To erase a file in CP/M, you use **ERA**; in Microsoft BASIC, use **KILL**. Despite its violent name, the command works in the same benign way as **FILES** does:

KILL "A:*.*" <cr>
KILL "YOREFILE.BAS" <cr>
KILL "B:*.BAS" <cr>

KILL does not support wild cards. It is used in the above example only as a safety feature.

The Microsoft BASIC equivalent of the CP/M **REN** command, **NAME AS**, works a bit differently:

BASIC	**CP/M**
NAME "OLDFILE" AS "NEWFILE"	REN NEWFILE= OLDFILE

We told you earlier that almost universally in computing, "new=old" is true, remember? We suppose Microsoft BASIC must be the exception that proves the rule, since it does just the opposite. Fortunately, the syntax of the command suggests the order in which the file names should appear.

The next command we'll introduce, **MERGE**, is often handy when you're creating large programs.

MERGE "programname" RETURN

MERGE takes the program file you specify and "merges" it with the program in memory. Where line numbering corresponds between the two programs, the program being merged from the diskette will replace the old program. Where line numbering diverges between the two programs, the new program will simply be added to the existing program. Keep two caveats in mind: the file to be merged must have been saved as an ASCII text file (see below), and if the **MERGE** command is used within a program, execution stops and you will see the **Ok** prompt again (this is not true if you use the **MERGE** option of the **CHAIN** command; see the Reference Guide for details).

Suppose you want to return to CP/M. Simply enter **SYSTEM** and press **RETURN**. Be careful with this command, however, because once you issue it, you have no way to recover any data your program may have stored in memory.

Earlier in this lesson, we promised you some more information about the MBASIC **SAVE** command and ASCII text files. When we first introduced the **SAVE** command, we did not tell you that three different formats enable you to save your programs:

ASCII text
binary
protected binary

"ASCII text" refers to a standard method by which your program saves every one of its characters on the diskette. You specify this option by appending a comma and an **A** to the **SAVE** command:

SAVE "filename",A RETURN

You can edit ASCII text files by using the **N** option in WordStar, but this type of file takes up more room on the diskette than the other types. ASCII, which stands for American Standard Code for Information Interchange, is a numbering system in which each different character has a different pattern of bits used to represent it; it is the format in which most computers store text.

Microsoft BASIC does not store your program in memory as ASCII text. Instead, it converts each command it recognizes into a single "special" character. For instance, it stores the command **PRINT** in memory as a 26, or 00011010 in the numbering method computers understand. This computer format is called "binary" because it uses the binary representation (so named because it involves counting by twos) to store each command. Text and variable names within a program are stored in ASCII format; but if you try to coax CP/M to print a copy of this file, all kinds of "weird" things will happen because of those binary commands. The binary format will be used to store your program files if you do not specify an option at the end of the **SAVE** command:

SAVE "program" RETURN

A third method to save programs is called "protected binary." This method uses a special "encrypted" form of the binary instructions, so that after the program has reloaded into the computer's memory, the program cannot be examined, only executed. DO NOT SAVE A PROGRAM IN THIS FORMAT unless you have copies in an unprotected format; otherwise, you'll never be able to make changes.

```
SAVE "program",P [RETURN]
```

We've come to the end of another section. By now you should be able to enter some simple BASIC programs, change them, store them on diskette, and retrieve them. We're now going to start introducing commands that build upon the base you already have. We strongly suggest that you try writing a simple program of your own without any additional instructions from us. When you pass this intermediate test, you'll want to plunge ahead. If you can't write a simple program yet, we suggest you review the material already presented before you continue.

Getting Information to a Program

If BASIC were restricted to the commands we've presented, it wouldn't be very flexible.

BASIC is an "interactive" computer language. Interactive means that BASIC presents some information, you interact with—or react to—it, and BASIC presents some more information. So far, we haven't shown you any ways—with the exception of typing ^C—to make BASIC pay any attention to what you're doing while a program is running; but we're now going to teach you how to do it.

The **PRINT** command sends information to your Osborne Executive's screen. To get information from the Osborne keyboard, BASIC uses the **INPUT** command:

```
INPUT variablename
```

If the type of the variable indicates that it stores a number—as in integer, floating point, and double-precision variables—BASIC will wait for you to type a number when it encounters the above statement. BASIC indicates it is waiting for you to enter something by displaying a question mark. If you enter something other than a valid number, an error message (**?Redo from start**) will show up, and the question mark will reappear. Press **RETURN** to tell BASIC that you've finished entering a value.

You can also employ string variables—variables that contain letters and symbols as well as numbers—in an **INPUT** statement. To indicate that you are finished entering information into a string variable, press **RETURN**.

A special version of the **INPUT** command combines the **PRINT** and **INPUT** functions:

```
LINE INPUT"What is your name?";N$
```

LINE INPUT is a special form of the **INPUT** statement designed to accept strings in their entirety. Unlike **INPUT**, the string may contain a comma in a **LINE INPUT** command. If you enter a comma in an **INPUT** string, MBASIC will think you're typing two strings, which causes an error. The **LINE INPUT** command cannot be used to input numeric variables; **LINE INPUT** also suppresses the question mark (?) prompt.

You can specify multiple variables by using a single **INPUT** statement:

```
INPUT NAM$,DATE$,AGE
```

In this case, you must type the information in the order the program expects it—name first, date second, age third as in our example—and you must separate each distinct item from the others by using a comma, not **RETURN**. Use **RETURN** to tell BASIC you're done entering all items. If you are going to employ this last form of the **INPUT** statement, make sure you use a message that explains exactly what users are to do; otherwise, they may not enter the information the way BASIC expects it.

```
10 PRINT"Sample Multiple Data Entry Program"
20 PRINT
30 PRINT"Enter the two pieces of information"
40 PRINT"requested below in the order requested,"
50 PRINT"each set off by a comma. Press RETURN"
60 PRINT"when done."
70 PRINT
80 INPUT "Enter NAME and AGE: ";NAM$,AGE
90 END
```

Note:

Lines 20 and 70 use a simple trick in BASIC—a ***PRINT*** *statement with nothing else on the line creates a blank line on the display because BASIC issues a RETURN when it comes to the end of a PRINT statement and doesn't find a comma or a semicolon telling it to stay at a location.*

What if you want to insert a comma into a string variable? You use a second variation of the **INPUT** command:

LINE INPUT"Enter CITY and STATE: ";CITYSTATE$

LINE INPUT accepts any character (up to 254 characters in all) in a string up to the terminating **RETURN**. If the words **LINE INPUT** are immediately followed by a semicolon—before the prompt message and variable name—pressing the **RETURN** key to terminate the input will not cause the cursor to move down a line, as normally happens when you press **RETURN**.

You now know how to get information and assign it to a variable. All of the methods so far described have the drawback that they require you to press the **RETURN** key when entry of information is complete. Even if you type only one character, you have to press two keys.

One way to cause BASIC to respond immediately to key presses is to use the **INPUT$** command:

> **CHARS$=INPUT$(NUMBER)**
>
> where **CHARS$** is the variable to store the input, and where **NUMBER** is the number of characters to expect.

As soon as BASIC receives the expected number of characters, program execution resumes with the next statement.

If you only want to check to see if a user has entered another character, use the following routine:

```
1000 'CHECK FOR CHARACTER ENTRY
1010 'CHAR=1 if character is ready, =0 if none
1020 '
1030         CHAR=0:CHAR$=""
1040         CHAR$=INKEY$
1050         IF CHAR$<>"" THEN CHAR=1
1060         RETURN
```

The key new statement here is the **INKEY$** command, which searches to see if anyone has entered a character. If so, **CHAR$** will contain that character upon execution of line 1040; if no character is ready, **CHAR$** will remain unchanged; we set it to nothing in line 1030.

The routine listed above is designed for use as a subroutine. Every time you wish to check to see if a character is entered, you add a **GOSUB 1000** command to your program. When the subroutine has executed and control has returned to the main portion of your program, the variable **CHAR** will have a value of **1** if a character has been entered, or **0** if no character is ready. You could then use the **IF** statement you learned earlier to conditionally execute portions of your program, depending on whether or not a character was typed.

Let's leave the subject of user-typed input and consider a different method of getting information to your program.

Wouldn't it be nice if you could "store" some data in a program and then look at it as you needed it? Well, you can. First, you need to store the data in **DATA** statements:

```
DATA 10,10,"Now, what is the",time,100
```

The above statement has five pieces of information:

information	**type**
10	**numeric**
10	**numeric**
"Now, what is the"	**string**
time	**string**
100	**numeric**

A comma separates each piece of data from the next. Notice that the third entry has a comma embedded in it and is surrounded by quotes. Use quotes to surround information destined for string variables only if you have a comma, an extra starting or ending space, or a colon in the data.

To use the information in a **DATA** statement, you must "read" it; the **READ** statement performs this task.

```
READ VALUE1,VALUE2,CHAR1$,CHAR2$,VALUE3
```

is a valid **READ** statement for the data in our example. You don't have to use all of the data in one **READ** statement. BASIC keeps track of how much of each **DATA** statement you've used, and will use the next unused data item each time it encounters a **READ**. The following is a valid method of using **DATA** and **READ:**

```
10 DATA Now, is, the, time
20 DATA for, all, good, people
30 DATA to, buy, Osborne, Executives
40 READ INFO1$,INFO2$,INFO3$,INFO4$
50 READ INFO5$
60 READ INFO6$,INFO7$
70 READ INFO8$,INFO9$,INFO10$,INFO11$,INFO12$
80 PRINT INFO1$+" "+INFO2$+" "INFO3$+" "+INFO4$;
90 PRINT " ";INFO9$+" "+INFO10$+" "+INFO11$+"
"+INFO12$
100 END
```

Note:

As has been our habit in this chapter, we have introduced a new concept in the context of a programming example. Note that lines 80 and 90 have "concatenated" (brought together) items to print. Try the example and note that a space is inserted between each word and the next when the word is displayed.

You can tell BASIC to forget where the next data item is located and to go back to the start of, or other location in, the list by using the **RESTORE** statement in your program. **RESTORE** by itself tells BASIC to start at the first **DATA** statement it encounters. You can also specify a line number following **RESTORE**, in which case the **DATA** statement beginning on the line number specified becomes the location of the next data items the program will use.

This lesson has taught you new things about getting information into the computer. We'll close the section by briefly introducing a few more items about getting information out of the computer.

The Osborne Executive stores 128 characters on each line and then shows you 80 of them. Therefore, every time BASIC detects that it is doing something past column 79, it "wraps around" to the next line. You can change this "default value" by using the **WIDTH** statement:

```
WIDTH NUMBER
```

where **NUMBER** is the value you want as the new screen width. The value that makes the most sense is 128. As long as your printer can print more than 80 characters on a line, you can also change the width of lines that appear on your printer by typing:

```
WIDTH LPRINTER NUMBER
```

Using the commands we've given you to work with so far, in order to display spaces on the screen you must enclose them within quotes in a **PRINT** statement. Two ways are available to circumvent this requirement:

```
PRINT TAB(NUMBER)
PRINT SPC(NUMBER)
```

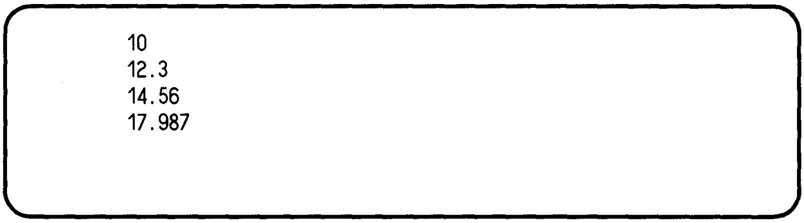
Either command will print the number of spaces you specify, and you can use either one in multiple-element **PRINT** commands:

```
PRINT "Now";TAB(30);"is";TAB(5)
```

We've saved the hardest command to master for the end of this lesson. If you've been experimenting while reading this chapter, you've probably noticed that it's not easy to line up the decimal places in amounts that contain different numbers of characters.

PRINT 10:PRINT 12.3:PRINT 14.56:PRINT 17.987

would appear as:



```
10
12.3
14.56
17.987
```

To make printed information appear the way you want it to, you use a formatting technique called a "mask." The mask is a string expression (it may be a variable) that contains a picture of the way you wish to format information. Here are the elements that go into the mask:

#	represents a digit position
+	represents the sign of a digit
−	represents trailing minus sign for negative numbers
**	means leading spaces will be filled with asterisks
$$	represents two digit positions and requests that a dollar sign appear at the left side of a number
&	represents a variable-length string position
\ \	enclose spaces to be printed
!	represents the first character of a string
^^^^	represents exponential notation
%	is a program flag to indicate that an oversize number has been printed
.	represents decimal point

Those descriptions probahly don't make much sense to you yet. Let's show you an example of how to print numbers up to $100,000.00 in the dollars-and-cents format:

PRINT USING "$$#####.##";VARIABLE

The **USING** in the above statement is what tells BASIC that the information in the following string will be used for the mask. Note that the information printed in the above example will have leading blanks (spaces) if the number doesn't totally fill the format string.

The number of combinations of the masking elements listed above is extraordinary, so you need to experiment with this feature to achieve the results you desire. Use the *Reference Guide* to help you further understand how each masking element works.

Advanced Variable Use and Functions

We're going to speed up the learning process a bit by using only brief introductions for most of the commands left for you to learn. We assume at this point that you are either a proficient programmer who is trying to learn the Microsoft BASIC syntax, or that you've consulted an outside work to learn more about BASIC and what it does.

If you're a newcomer to computing and have made it this far without outside help, GREAT! We are not the final authorities on everything, however, and we think some additional help will facilitate your learning.

You're continuing, eh? Okay, buckle up; the pace will be quicker.

Microsoft BASIC has built-in functions, much like the function keys on some hand calculators. These functions are often a shorthand way of performing an action or calculation that otherwise might take considerable programming effort.

Let's briefly examine the functions available:

ABS(NUMBER)	the absolute value of the number
ASC(CHAR$)	the ASCII value of the first character in a string
ATN(NUMBER)	the arc tangent of the number
CDBL(NUMBER)	converts number to a double-precision number
CHR$(NUMBER)	the character string that has the ASCII value of the number
CINT(NUMBER)	converts number to an integer by rounding
COS(NUMBER)	the cosine of the number
CSNG(NUMBER)	converts number to single-precision number
EXP(NUMBER)	raises e to the power of the number
FIX(NUMBER)	the truncated portion of the number (not rounded)
FRE(NUMBER)	the amount of free memory available
HEX$(NUMBER)	the string in hexadecimal representation of the number
INT(NUMBER)	the rounded-down integer of the number
LOG(NUMBER)	the natural logarithm of the number
LPOS(NUMBER)	the current position of the printer's printhead
OCT$(NUMBER)	the string in octal representation of the number
PEEK(NUMBER)	the value stored at the memory location with the address of the number
POS(NUMBER)	the current position of the cursor

RND(NUMBER)	a random number between 0 and 1
SIN(NUMBER)	the sine of the number
SPACE$(NUMBER)	a string of spaces equal in length to the number
SQR(NUMBER)	the square root of the number
STR$(NUMBER)	the string equal to the value of the number
TAN(NUMBER)	the tangent of the number
VAL(CHAR$)	the numerical value of the string

The method by which you use these functions is to assign them to a variable:

VARIABLE=FUNCTION(PARAMETER)

Make sure that the type of variable you use matches the type of function you're using—string variables should be used with string functions, for example.

A second set of functions you'll want to know about are some special string functions:

LEFT$(CHAR$,NUMBER) — used to get the string of characters of the number's length from the left-hand side of a string you specify.

RIGHT$(CHAR$,NUMBER) — used to get the string of characters of the number's length from the right-hand side of a string you specify.

MID$(CHAR$,STARTPOS,NUMBER) — used to get the string of characters of the number's length from the starting position in the middle of the string specified.

INSTR(STARTPOS,CHAR1$,CHAR2$) — used to get the position (number) at which the occurrence of the second string is duplicated by the first string.

If you need more information about any of these functions, turn to the Reference Guide for a complete description of their use.

For the most part, we've presented the straightfoward aspects of Microsoft BASIC. It's now time to examine some of the "sneakier" commands.

DIMENSIONING VARIABLES: The term "dimensioning" refers to the process of telling BASIC ahead of time how much space to reserve for a variable. You must do this when you use a variable as an array: a table of values all referenced by the same name. Say you wanted to maintain a table of values consisting of four columns and four rows of information. You could do so by assigning the following variable names:

VALUE1	**VALUE2**	**VALUE3**	**VALUE4**
VALUE5	**VALUE6**	**VALUE7**	**VALUE8**
VALUE9	**VALUEA**	**VALUEB**	**VALUEC**
VALUED	**VALUEE**	**VALUEF**	**VALUE0**

This could become unwieldy, however, especially if you had a large table. In BASIC, you could create this table like this:

```
OPTION BASE 1
DIM VALUE(4,4)
```

Your table would now look like this:

VALUE(1,1)	**VALUE(1,2)**	**VALUE(1,3)**	**VALUE(1,4)**
VALUE(2,1)	**VALUE(2,2)**	**VALUE(2,3)**	**VALUE(2,4)**
VALUE(3,1)	**VALUE(3,2)**	**VALUE(3,3)**	**VALUE(3,4)**
VALUE(4,1)	**VALUE(4,2)**	**VALUE(4,3)**	**VALUE(4,4)**

Mathematicians will immediately recognize this concept as a "matrix." The numbers in the parentheses following the variable name are simply the location of that item in the matrix. In BASIC, we call this concept an "array," and this particular array is a two-dimensional one.

But what about that **OPTION BASE 1** statement—what is it doing? Most computers use internal counting procedures that begin with zero. The **OPTION BASE** statement we use before the **DIM** statement sets the internal counting BASIC uses to begin at one.

You must declare all multidimensioned variables by using a **DIM** statement before you attempt to use them in your program. The maximum number of dimensions in an array is 255, and the maximum number of entries in any one dimension is 32767. Dimensioning does take up memory space, so be careful not to get carried away with this concept, or else you might find you have little or no room to expand your program.

You can make BASIC "forget" your dimensioning commands by using the **ERASE** command followed by a list of variable names. If you do not use this statement and try to redimension a variable, you will get an error message.

Another method of quickly starting over with variables is to use the **CLEAR** command:

```
CLEAR ,topofmemory,stackspace
```

Topofmemory is the topmost memory address BASIC should use, and **stackspace** is the amount of stack space (temporary information storage) you wish Microsoft BASIC to use. All numeric variables are set equal to zero, while string variables are set to "null" ("" indicates "nothing" between the quotes) when you use the **CLEAR** command.

DEFINING: You can assign a variable type to all variable names beginning with a single letter or range of letters by using the **DEF** statements in Microsoft BASIC:

DEFINT A	defines all names beginning with A as being integer variables.

DEFSTR B-C defines all names beginning with B and C as being string variables.

DEFSNG A,Z defines all names beginning with A and Z as being single-precision variables.

DEFDBL A-Z defines all variable names as being double-precision variables.

In general, Microsoft BASIC deals with integer variables the most quickly, and with double-precision and string variables the most slowly. Your programs will execute more quickly if you use integer variables for **FOR/NEXT** loops and other control structures where possible.

OTHER VARIABLE TRICKS: Let's say that you have two variables and want to exchange their values. In most BASICs, you effect the exchange by entering something like this:

```
5  'Initial values
10 X=10
20 Y=20
30 'Now swap 'em
40 Z=X:'use dummy variable temporarily
50 X=Y
60 Y=Z
```

Keeping track of that third, "dummy," variable is sometimes difficult, and it certainly masks your real intent: you don't want to assign a value to Z; you just want to swap X and Y. Microsoft BASIC lets you use one statement to perform the swap:

SWAP variable1,variable2

The only limitation of this command is that the variable types must match.

Advanced Control Structures

You already know about the **IF/THEN** structure introduced earlier. Let's add some new variations to the control structures you know.

One method of extending **IF/THEN** is to put multiple statements to the right of the test:

```
IF test THEN statement1:statement2:statement3
```

Each statement following the **THEN** will execute, in order, only if the test condition is true (unequal to 0).

You can also extend the **IF/THEN** construct by using the **ELSE** option:

```
IF test THEN statement1 ELSE statement2
```

This structure is useful if you wish to program an "either/or" situation. In the above example, if the test proves "true," then "statement1" executes; otherwise, "statement2" executes.

What if you want to test for more than two conditions (true and false)? Simple—use the **ON/GOTO** or **ON/GOSUB** construct:

```
ON value GOTO place1,place2,place3, etc.
ON value GOSUB place1,place2,place3, etc.
```

The "value" encountered directs execution to the "place" that corresponds to that value. So, if the value is 1, the first line number named ("place1") will be used; if the value is 2, the second line number named is used, and so on. When you use **ON/GOTO** or **ON/GOSUB**, make sure you have a line number to go to for every possible value you can encounter. If the value is greater than the number of line numbers listed, execution will drop down to the statement following **ON/GOTO** or **ON/GOSUB**. Most programmers put a "trap" either immediately preceding or immediately following the **ON/GOTO/GOSUB** construct so that an unexpected value is captured and reported rather than allowed to "wander off in the program untended."

The last control structure we'll examine is the **WHILE/WEND** construct. Here is the basic layout of this structure:

```
WHILE test
STATEMENTS
WEND
```

The **STATEMENTS** in the body of the **WHILE/WEND** loop will execute over and over, in order, until the test is equal to zero. This loop structure is similar to the **FOR/NEXT** loop. Unlike the **FOR/NEXT** loop, however, the **WHILE/WEND** structure executes a variable number of times. It is important to make sure you've provided a way for the test to become false, because you may otherwise find that you have coded another "endless loop."

Disk Files and BASIC

Even with all the fancy and complicated BASIC commands you've learned, Microsoft BASIC wouldn't really be too useful without a permanent method for storing and retrieving information your programs use. One reason we've saved this section for last is that the way Microsoft BASIC "talks to" disk files is not intuitively obvious to casual observers.

Two methods exist to store information on a diskette and retrieve it: sequential access and random access. A file you maintain using "sequential access" is one in which you must always start with the first piece of information in the file and work your way to the item you want. Watching TV is a sequential-access activity—even if you want to watch only the last five minutes of a program, the first 25 minutes must still play.

Take an example. Say you are storing the scores of your five favorite football teams on a diskette. One week's scores might be:

Team 1	**45**
Team 2	**14**
Team 3	**20**
Team 4	**21**
Team 5	**32**

You might store this in a sequential access file like this:

1,45,2,14,3,20,4,21,5,32

The format in the above example is team number, score, team number, score, and so on. To find the score for team 5, you must first "read" all the rest of the information in the file that precedes it.

Each piece of data in a sequential-access file is separated from the others by a comma. Let's examine the process necessary to create a sequential-access file:

1. "Open" the file to be used (ready it for use).
2. "Write" information to it.
3. "Close" the file when you finish adding information.

Each process above has a BASIC command associated with it. Using our football-score example, the following program statements would create a sequential-access file:

```
100 '  CREATE A FOOTBALL SCORE SEQUENTIAL-ACCESS
       FILE
110 '
120    OPEN "O",#1,"SCORES.DAT"
130    PRINT#1,1,45,2,14,3,20,4,21,5,32
140    CLOSE #1
```

The format of the **OPEN** statement consists of the following information:

- the word OPEN;
- O to write information to a file, or I to get information from a file;
- an arbitrary number assigned to that file (#1 here); and
- the name of the file to be used.

The file has an assigned number so you can have multiple files all open for use at the same time. Since each file has a different number associated with it, Microsoft BASIC knows which file you wish to write information to.

In fact, line 130 in our example shows how you specify how to write to a sequential-access file. Note that—except for the **#1**—the write-to-file statement takes the same form as the **PRINT** statement. When Microsoft BASIC sees a number sign followed by a number in a print statement, it assumes that you are referencing a file and checks to make sure it is open for use.

Okay, you now know how to create and write information to a sequential-access file—how about getting information out of it? Simple—you can substitute the following lines in the above program:

```
120  OPEN "I",#1,"SCORES.DAT"
125  FOR LOOP=1 TO 5
130       INPUT #1,TEAM,SCORE
135       PRINT TEAM, SCORE
137  NEXT LOOP
```

The lines above read a team number and score from the file and display them on your screen.

Next you need to know how to add information to a sequential file . . . it's not as easy as you may think. To add information, follow these steps:

1. Open the file to you want to add to in the I (for input) mode.
2. Open a second file with a "dummy" name in the O (for output) mode.
3. Read the data from the first file and write it to the second.
4. Add your new information to the second file.
5. Close the second file.
6. Close the first file and use the **KILL** command to delete it.
7. Rename the second file using the **NAME AS** command. Make sure that it ends up with the same name as the first file (the one you were adding to).

This procedure can be complicated, and the limitation of disk-storage space available will compound your problems if you are maintaining a lot of data in a sequential-access file.

Two functions sequential-access files often use are **EOF** and **LOC**.

EOF(filenumber) is true if the next item in the file is the file's end and false if you're in the middle of a file.

LOC(filenumber) gives the number of sectors read or written to in a sequential file.

Use these functions just as any other function we described earlier.

Random-access files may require a little extra effort to learn; but once you've written a few programs using random files, you'll find that the commands come naturally to you.

In a random file is one in which everything is ordered in such a way that you can go directly to a piece of information you wish to read or write. If watching TV is somewhat akin to reading a sequential file, using a videotape recorder to find a particular frame of a TV show is akin to reading a random-access file.

Let's continue to use our example of five football teams and their scores from last week's games. What if every team number consisted of one digit, and every score consisted of two digits? Every combination of team and score would occupy three characters in our file. To get to team three's score, we'd then need to tell the computer to go to the eighth and ninth characters in the file and read them. So we'll find something like this in the file:

```
145214320421532
```

That mishmash isn't very easy to scan, but that's okay because you're going to make the computer scan it. Before you continue, however, make sure that when you first set up a random-access file you allow for all possibilities. You must consider what will happen if you have a score less than 10 or greater than 99. So that your program can accommodate such an eventuality, you should make team numbers two digits in length and use zeros in front of single-digit numbers. You might also want to make the scores three digits long if you expect prodigious point production.

Let's create our random file. In this example, we're going to use comments on every line to explain what is happening.

```
100 '  CREATE A RANDOM DISK FILE OF TEAMS AND SCORES
110 '
120    OPEN “R”,#1,“SCORES.DAT”,4:'note “R” and number at end
130    FIELD #1,2 AS TEAM$,2 AS SCORE$:'we define fields in file
140    FOR LOOP=1 TO 5
150       READ TEAMNUM$,TEAMSCORE$:'scores are in a DATA
          statement
160       LSET TEAM$=TEAMNUM$:'get information into field
170       LSET SCORE$=TEAMSCORE$:'get information into field
180       PUT #1,LOOP:'store it on disk
190    NEXT LOOP
200    CLOSE #1
210    DATA 01,45,02,14,03,20,04,21,05,32
```

The **OPEN** statement for a random file is different in two respects from the sequential-access **OPEN** statement: first, you use **R** (for "random") instead of **I** or **O**; second, after the file name you must add a comma and tell Microsoft BASIC how long each "record" in the file will be.

A "record" is one complete set of data. It's kind of like a file folder in a file drawer. Within the record can be a number of "fields." A field is one individual set of data, like a sheet of paper in a file folder. Here's an example of one file hierarchy:

file:	Osborne owners' names and addresses	"OZ.DAT"
record:	Information on one Osborne owner	RECNUM%

fields:	Osborne owner's name	NAM$
	Osborne owner's address	ADDRESS$
	Osborne owner's city	CITY$
	Osborne owner's state	STATE$
	Osborne owner's ZIP code	ZIP$

We have put the variable names that reference each item in the right-hand column.

Next in the sample random-disk-access program comes the **FIELD** statement. Given the information we just presented, you should guess that you have to identify each field and give it a length. The total length of the fields must add up to the total length specified in the **OPEN** statement.

To write information into a random-access file, you first must move the information into the variables named in the **FIELD** statement, using either **LSET** (for leftset, or left justify) or **RSET** (for rightset, or right justify). Any information that "spills over" the amount of room set up for a given field will vanish.

Note:

DO NOT USE THE VARIABLES NAMED IN THE FIELD STATEMENT FOR ANYTHING EXCEPT MOVING INFORMATION TO AND FROM THE DISKETTE. If you use one of these variables to display something on the screen or to change a value, you will confuse Microsoft BASIC's internal buffer "pointers" and get some interesting—but incorrect—results.

Finally, to actually write the information onto the diskette, use the **PUT** statement. Its two elements are the file number you'll use and the record number where you want to store the information. A nice feature of random-access files is that you can begin by storing information in record number 50, if you want, and then come back and fill in the rest at a later time.

To get information from a random-access disk file, simply get rid of the **LSET** or **RSET** statements and use **GET** in place of **PUT**. After your **GET** statement, you'll want to move the information from the field variables to variables you can use within your program:

```
120   OPEN "R",#1,"SCORES.DAT",4
130   FIELD #1,2 AS TEAM$,2 AS SCORE$
140   FOR LOOP=1 TO 5
150           GET #1,LOOP
160           TEAMNUM$(LOOP)=TEAM$
170           TEAMSCORE$(LOOP)=SCORE$
180   NEXT LOOP
190   END
```

We keep using string variables in our examples to store numeric information because Microsoft BASIC stores all information on disk in a special binary form. We're using strings so you don't have to deal with the conversion from the binary form to the one used within your program.

To convert numbers into their requisite format for storage in random-access files, you must use one of the following commands to do the conversion in conjunction with the **LSET** or **RSET** command:

LSET(FIELDVARIABLE)=MKI$(VARIABLE) for integer variables
LSET(FIELDVARIABLE)=MKS$(VARIABLE) for single precision
LSET(FIELDVARIABLE)=MKD$(VARIABLE) for double precision

To restore the information to the form you can use within your program, use the following conversion commands instead of the **LET** statement used in the above read-from-random-file example:

VARIABLE=CVI(FIELDVARIABLE) for integer variables
VARIABLE=CVS(FIELDVARIABLE) for single precision
VARIABLE=CVD(FIELDVARIABLE) for double precision

We want to tell you only one more thing about files, and that is how to change diskettes and let Microsoft BASIC know that you've done so. Use the **RESET** command. Simply placing **RESET** in your program at the point immediately before you remove the first diskette from the drive closes all files in use and rewrites the correct information about them in the CP/M directory onto that diskette.

Closing Up the BASIC Shop

A few commands we have not covered in this text are fully explained in the *Reference Guide*:

CALL PEEK POKE INP OUT
DEF USR WAIT DEF FN USR &H
&O RANDOMIZE VARPTR ERR ERL
ERROR ON ERROR NULL RESUME /F:
/M: /S:

We hope we gave you a good "taste" for Microsoft BASIC.

Index

WordStar

Index

SuperCalc

Index

CBASIC and MBASIC

NOTES

NOTES

NOTES

OSBORNE™
COMPUTER CORPORATION

3FØØ194-ØØ