SH20-9060-2

Program Product

VS BASIC for VSPC: Terminal User's Guide

IBM

SH20-9060-2

Program Product

Program Number 5748-XX1

IBM

Third Edition (September 1978)

This edition replaces the previous edition (numbered SH20-9060-1) and makes it obsolete.

This edition applies to Release 3 of VS BASIC, program number 5748-XX1, and to any subsequent releases unless otherwise indicated in new editions or technical newsletters.

Information pertaining to Release 2 of VSPC is for OS/VS2 MVS and OS/VS1 only; for Release 2 of VSPC under OS/VS1, information is for planning purposes only until its availability.

The changes for this edition are summarized under "Summary of Amendments" following the list of figures. Specific changes made are indicated by a vertical bar to the left of the change. These bars will be deleted at any subsequent republication of the page affected. Editorial changes that have no technical significance are not noted.

Information in this publication is subject to significant change. Any such changes will be published in new editions or technical newsletters. Before using the publication, consult the latest *IBM System/370 Bibliography*, GC20-0001, and the technical newsletters that amend the bibliography, to learn which editions and technical newsletters are applicable and current.

Requests for copies of IBM publications should be made to the IBM branch office that serves you.

Forms for readers' comments are provided at the back of the publication. If the forms have been removed, comments may be addressed to IBM Corporation, P. O. Box 50020, Programming Publishing, San Jose, California 95150. All comments and suggestions become the property of IBM.

PREFACE

This publication provides the tutorial and reference information necessary to develop programs in the VS BASIC language under one of the VS Personal Computing program products:

- VSPC OS/VS1 (Program Number 5740-XR5)
- VSPC OS/VS2 MVS (Program Number 5740-XR6)
- VSPC DOS/VS (Program Number 5746-XR3) (Release 1 only)

The material in this guide covers the following information necessary for a typical user to operate VS BASIC under VSPC:

- The structure of VSPC, including workspaces, libraries, filenames, and passwords.
- Operation of terminals with the assumption that the user is typing on an IBM 3270 Information Display System, an IBM 2741 Communication Terminal, or on an IBM 3767 Communication Terminal in start/stop mode. (A chart outlining essential information for all VSPC terminals is in Appendix D.)
- Techniques of logon, error correction, and logoff.
- Command language for creating, modifying, and running programs.
- Library maintenance, controls, file organization, data sharing, messages from the systems, and a sample terminal session.

Five appendixes are included:

- Device dependencies.
- Information on how to convert other data files to VSPC files for use with VS BASIC.
- A brief discussion of batch processing.
- A chart with the basic steps necessary to communicate with VSPC on any of six types of terminals.
- A summary list of VSPC commands used in VS BASIC.

The reader is assumed to be familiar with the VS BASIC language as explained in the publication *VS BASIC Language,* GC28-8303, which is a prerequisite to this guide.

References

VS Personal Computing (VSPC) General User's Guide and Command Language, SH20-9071

VS Personal Computing (VSPC) Terminals, SH20-9073

VS BASIC Language, GC28-8303

VS BASIC Installation Reference Material, SC28-8309

VS BASIC for VSPC: Reference Summary, SX26-3710

CONTENTS

FIGURES

SUMMARY OF AMENDMENTS

September, 1978

VSPC RELEASE 2 (for OS/VS1 and OS/VS2 MVS)

New Programming Features

Support for additional display system devices

Improved terminal support
VIEW mode (full screen) editing
Program function keys

Error message clarification

Current line pointer support

New commands: PFKEY, VIEW, COPY, LOCATE, EXTRACT, JOIN, SPLIT, TEXT, RELEASE, ACQUIRE.

Improved commands: LINESIZE, TRANSLATE, HARDCOPY, INPUT, LIST, CHANGE, FILE, PUNCH.

December, 1976

Release 3 of VS BASIC

The following subset of Release 3 enhancements for VS BASIC are discussed in this publication:

- A new OPTION statement can affect the specification of the VSPC ENTER command because the statement can override precision specifications of the command.
- A new ON statement can affect the use of the attention key because the statement permits the VS BASIC program to control the response to an attention signal and can trap some error for program processing during program execution.
- Two new statements, PRINT TO and INPUT FROM, are provided. These statements allow data that would normally be directed to or from a terminal to be directed to or from a user-specified file.
- The REUSE clause has been added to the OPEN statement. This clause enables you to reuse existing files.
- The ability to access VSPC direct files is now available to the VS BASIC user through VS BASIC's relative-record capability.
- A new REM clause allows comments to be affixed to most VS BASIC statements.
- Some operators can be represented by mnemonics in addition to their symbolic representation.

Miscellaneous editorial changes have been made to clarify sections of this publication.

August, 1976

VSPC AID

New Programming Feature

A description of VSPC AID has been added to the section on VSPC commands.

Specification Changes

Response to Input Request

A note has been added to the entry on "Conversing with VSPC"

DESCRIPTION OF VSPC

VS Personal Computing (VSPC) makes a computing system available to many terminal users at the same time. Scientific computation, computer programming, and information retrieval are all possible to VSPC users through communication terminals.

The problem-solving nature of VS BASIC and the interactive nature of VSPC are particularly well suited to each other. Together, they provide a quick and easy means to help you get your work done.

VSPC, which runs under OS/VS1, OS/VS2 MVS, and DOS/VS (Release 1 only) offers a set of functions tailored for users without extensive data processing knowledge. VSPC has commands (instructions) that let the time-sharing user create and execute VS BASIC programs from a special typewriter or display station called a terminal. With the commands available, you can:

- start and end terminal sessions
- format your work
- create programs
- modify programs
- execute programs
- maintain libraries
- send and receive messages

VSPC is conversational; it interacts with you. It executes your commands, lets you know when input is needed, and keeps you informed of what the computer is doing. Because it is conversational, you can develop programs at your terminal instead of on paper, so that developing and maintaining your programs is straightforward and easy. In VSPC, interactive processing (online processing) takes place in an area known as the *foreground.* Foreground processors, like VS BASIC, are designed to make use of the VSPC interactive facilities.

Interactive processing contrasts with conventional *batch* processing. In batch processing, you tell the computer system to perform a series of actions in one complete set of instructions. The computer then tries to execute the whole series, and if it can't perform some or all of your requests, it records one or more error messages. The results of the entire series of requests—and the error messages—are then returned to you as a single unit. Each batch job is processed as time permits; thus, the results may not be returned to you for some time. As you can see, such processing cannot be conversational. Batch processing (offline processing) in VSPC takes place in an area known as the *background.* The VSPC job entry facilities let you submit VS BASIC programs as batch jobs.

VSPC allows you and a number of other terminal users to do conversational processing at the same time. That is, you can all talk to VSPC and get VSPC's answers simultaneously.

Because VSPC must have a way of identifying your work, you are assigned a *user number.* When you log on to VSPC (that is, when you begin a terminal session), you must always identify yourself with this user number.

Your user number is assigned by the *VSPC administrator,* a user who has the responsibility for the VSPC facilities available to you. The VSPC administrator assigns your user number and specifies other characteristics of the work you'll be doing.

When you log on, VSPC furnishes you with a *workspace.* This *workspace* is an area you can use much as you use a blackboard or scratch pad in ordinary problem solving. You can write down solutions, erase them, modify them, add to them, subtract from them, write in reminders to yourself. If the problem you are working on needs only a one-time solution, you can execute it, and then erase it from your workspace. Alternatively, if the problem solution will be needed again later, you can copy the solution and save it for future reference.

You save the copy in a user *library.* When your user number is assigned, VSPC automatically assigns you at least one such library, and may make more than one available to you. VSPC uses one general library that contains a library for each user. In your library you save *files,* each identified by a *filename.* VSPC keeps a directory of filenames for the files in all libraries.

In your files you can place *data. Data* can be anything you want it to be. It can be a specific problem solution, information you will use to test your problem solution, lists of computer instructions or statements, or a set of random numbers or letters.

When you enter *data* from your terminal (type it in from the terminal keyboard) you are placing the *data* in your workspace. Once the *data* is there, you can edit it—add lines, delete lines, delete characters, move lines from one area to another, change words and phrases, and so forth. You can print or display it at the terminal, save it in a library, and/or erase it.

If your data should not be used by anyone else, you can specify a *file password,* known only by you, which must be supplied before the data can be retrieved from the library. You can assign a *file password* to each of your files. In addition, your VSPC administrator may assign you a *logon password* which you must supply when you log on to do work; this is to prevent unauthorized access to your library. You can change either kind of password whenever necessary.

You do all your work through VSPC *commands.* Commands tell VSPC to perform specific actions. There are commands that help you use the terminal (for example: tell the computer what terminal tab settings to use), commands to edit your data, and commands to manipulate your workspace (for example: erase any data in it). There are commands to manage your library files, do batch processing, and perform other actions.

Your terminal input consists of two kinds of information. *Data* is material you type that you want processed, such as VS BASIC programs. *Commands* manipulate this data in the creation of VS BASIC programs.

Data

Data can be source programs, command lists, prose, or numbers. Data entered into VSPC from your terminal is distinguished from a command by the fact that a line number begins the line. For example:

```
30 The cat is white
```

is treated by VSPC as data and placed in your workspace. However, if you type

```
The cat is white
```

the system responds:

```
UNKNOWN COMMAND 'THE'
```

because there is no line number and VSPC assumes you are typing a command that it does not recognize.

- If you type

```
30 save
```

expecting to have your program saved, the system accepts your "command" as data and places the line in your workspace.

Line numbers are limited to five digits. You can type the line numbers yourself, or you can request VSPC to generate line numbers for you. See "INPUT Command."

Commands

Commands generally consist of a descriptive word that specifies a particular action to be performed (MOVE, CHANGE, SAVE) followed by an optional *operand* that tells VSPC what to move, change, or save. For example:

```
move 30:70 120
```

tells VSPC to move lines 30 through 70 in your workspace to a position following line 120.

Each command tells VSPC to perform a specific action. Commands can be grouped by processing capability:

Terminal Oriented Commands: let you log on and log off (start and end a terminal session), change your logon password, change the line length and tab settings you send to VSPC, and change lines you have already entered. On the IBM 3270 you can set the program function keys to often-used commands or character strings.

Message Commands: let you send messages to users on other terminals, and prepare your own terminal to accept or refuse messages from others.

Data Editing Commands: let you add or change a line in the workspace, delete characters, delete lines, replace lines, move or copy lines up or down, split or join lines, change line numbering, and locate occurrences of specific groups of characters. You can specify whether or not you want certain commands to display the text of the affected line at your terminal.

Workspace Oriented Commands: let you change the *attribute* of your workspace, name your workspace contents and save them in a library, and erase the workspace contents when they are no longer needed.

Library Management Commands: let you add and remove files in a library, transfer files among users, specify that other users can or cannot read a file, and specify that the file cannot be changed. You can also display—that is, print or obtain some other visual presentation—the library directory.

Job Entry Commands: let you submit jobs for batch processing, to query their status, to cancel them, to specify that job output is to be sent to some specific destination, and to erase job output once it has been produced. (Job entry commands are not always available for your use; check with your VSPC administrator.)

AID Commands: let you receive online explanations of VSPC messages, and online prompting in the formation of valid VSPC commands. (AID commands are not always available for your use; check with your VSPC administrator.)

Abbreviation of Commands

You can abbreviate VSPC commands and keyword operands to the first three characters, and the abbreviation is valid. (The one exception is the logon command (**VSPC ID=***usernum*); check with your system administrator whether or not VSPC can be abbreviated.)

You can shorten some command names to fewer than three characters if they cannot be confused with any other command name. Correct characters up to the full length of the word are accepted. For example, the CLEAR command is understood by VSPC if you type CL, CLE, CLEA, or CLEAR.

Keywords can be abbreviated to fewer than three characters if they cannot be confused with any other keyword within the command. For example, NOPROMPT in the INPUT command is understood by VSPC if you type N, NO, NOP, NOPR, NOPRO, NOPROM, NOPROMP, or NOPROMPT.

VSPC will interpret either of the following examples in the same way:

```
send 'message-text' operator
se 'message-text' o
```

In this book, abbreviated examples have a minimum abbreviation of three characters.

Separators in Commands

A separator is a character that separates words or values in a line of input. Two rules apply to their use in VSPC.

- The separators are the blank, the comma, and the tab.
- Multiple separators are considered one.

Command Punctuation

Punctuation in operands has special meanings:

- Parentheses enclose variable information.

  Note: Parentheses may be omitted unless their omission makes the command ambiguous, such as when changing a file password with the PROTECT command.

- A colon (:) placed between two numbers specifies a range of numbers.

- Single quotation marks (') precede and follow groups of characters—called character strings—that are to be treated by VSPC exactly as specified.

Command Continuation

When the command *entry* ends with a minus sign (-), the next succeeding line is treated as a command continuation; logically, it immediately follows the last character preceding the minus sign. (Note that separators are considered to be characters.) For example, the following terminal entry:

```
query-
  file(filea)
```

is accepted as if entered as:

```
query      file(filea)
```

When the command *entry* ends with a plus sign (+), the next succeeding line is treated as a field continuation; logically, the first nonseparator character of the succeeding line immediately follows the last character preceding the plus sign. For example, the following terminal entry:

```
query file+
  (filea)
```

is accepted as if entered as:

```
query file(filea)
```

Command Entry

You can type in either lowercase or uppercase letters in VSPC. The one exception is the logon command. VSPC must be typed in either all uppercase or all lowercase letters. You can mix upper and lowercase letters in all other commands, but you will find it most convenient to type in lowercase whenever possible. (VSPC interprets all alphabetic characters in command names or keywords as uppercase.)

For clarity of presentation, the examples and syntax presentations in this book show your entries in lowercase and system responses in uppercase alphabetic characters.

Definition of Terms Used in Command Descriptions

- An *operand* is a word following a command name that modifies the processing of the command. Some operands are required; some are optional.
  a. Keyword operands define a predetermined action to VSPC; keyword operands are shown in uppercase letters. For example, CONTINUE in the OFF command.
  b. Variable operands represent information supplied by you, the user, such as *message-text, user number, password, line-number, string, incr* (increment), or *filename.* Variable operands are shown in lowercase italics.
- A *default* is the operand that the command assumes if none is specified; defaults are noted in the command representation and in the text.

VSPC AID

If VSPC AID has been included at installation time, you can ask the program to help you specify the operands for VSPC commands correctly. For any VSPC command, enter the command name preceded by a question mark (`?`). VSPC AID will reply with a series of prompts that question you about the operation you want to perform. From the replies, AID forms the command in valid format and then allows you to choose whether or not to execute it.

VSPC AID will also display an explanation of error messages sent by VSPC. To display an explanation of the *n*th preceding message, enter `?n`.

If you have forgotten the proper name of the command you want to specify, you can get a categorized list of VSPC commands by entering `?COMMANDS`. To display an online explanation of all the capabilities of VSPC AID, enter `?AID`.

VSPC Character Sets

The characters used in VSPC commands are:

- Alphabetic characters—29 alphabetic characters: 26 letters (uppercase or lowercase) plus three alphabetic extenders ($, #, @)
- Digits—Ten digits (0, 1, ..., 9)
- Special characters—( ) + * - / ? : ' = for punctuation and command formats

The characters that VS BASIC recognizes are listed in the publication *VS BASIC Language*.

User Profiles

Before VSPC can recognize you and allow you to do work, your VSPC administrator (a VSPC user assigned the task of administering VSPC) must introduce you to VSPC through a *user profile*. Your user profile contains a definition of the characteristics associated with your use of VSPC. It includes:

- a user number that identifies you to VSPC
- a workspace attribute that tells VSPC what kind of work you'll most often be doing
- a logon password that you must supply each time you log on; having a password is optional, and you can create, change, or delete one through the PASSWORD command
- the type of library assigned to you: private, project, or public (see section, "Libraries")
- whether or not you have access to a project library
- the maximum storage allocated for your library and workspace
- a job entry code that tells VSPC whether or not you are able to use job entry facilities
- other system-related information

Except for your logon password, you can display your profile with the QUERY command.

Any questions you have concerning your profile should be directed to your VSPC administrator.

Workspace Attributes

One of the characteristics described in your user profile is the workspace attribute that the system assumes if you do not designate one. The attribute of your workspace names the kind of contents VSPC expects your workspace to contain unless you specify another one. Your workspace attribute will probably be BASIC or LBASIC (denoting short or long precision numbers), but you can also use the DATA attribute to create data files to be used with your VS BASIC programs, or the CLIST attribute to create and execute command lists. (A command list is a series of VSPC commands. See "Command List (CLIST) Processing".)

When you log on to VSPC, the workspace attribute is determined by the CONTENT stated in your user profile. After you log on, you can issue the ENTER command to change the workspace attribute.

When you load or run a file, your workspace assumes the content attribute of that file.

Libraries

In addition to the workspace, VSPC provides libraries into which you place the data in your workspace (as *files*) until you want to work on that data again.

When the VSPC administrator creates your user profile, he makes available to you a library using your user number as your library identification number (libnum). You are defined as the *library manager* of this library.

When you save new work in any library, you become the *file owner* of every file you create, and you must always specify a unique filename for it. (See "Filenames" section.)

In any library, only the *file owner* can modify a file he has created. Only the *file owner* and the *library manager* can delete it.

There are three kinds of libraries: private, project, and public, each with different levels of accessibility. You may be the library manager of any kind.

Private Libraries

A *private* library is normally available only to the user whose user number matches the library number. You can, if yours is a private library, specify, through the SHARE command, that specific files can be accessed by other users. (The SHARE command cannot be issued for the special file CONTINUE.)

If yours is a private library, your VSPC administrator can make a project library available for your use.

Project Libraries

If your library is a project library, then a specific group of VSPC users is allowed to read files from your library. Your VSPC administrator can define your project library in either of the following ways:

1. Noncontrolled—Other project users are allowed to place files in the library; each such user the *owner* of the files he places within it. You are the *library manager;* however, you are the *file owner* only of those files you yourself place in the library. Only the owner of a file can modify it; however, ownership of a file can be transferred from one project user to another. Only the owner or you, the library manager, can remove it.
2. Controlled—Other project users can read files in the project library; they cannot place files into it. Thus, you are the owner of every file, and so only you can modify or remove anything in the library. Such libraries are known as controlled project libraries.

If yours is a project library, your VSPC administrator can also make another project library available for your use.

You can refer to your project library with a user number of 0 (zero).

Public Libraries

If your library is a public library, then every VSPC user can read files from your library. Public libraries can be of the same two kinds as project libraries, and the same rules apply to file modification and removal, except that ownership cannot be transferred.

If yours is a public library, your VSPC administrator can also make a project library available for your use.

File Types

The VSPC user has access to five types of files: sequential, direct, object program, undefined, or external. Each type is described in the following paragraphs.

Sequential Files: are ones in which your VS BASIC programs will always process the records in the order of their line numbers. The records are placed in the file in line number order, and your VS BASIC programs will always retrieve them from the file in that order. Records can be of varying lengths, and the increment between line numbers is of no importance. Using VSPC commands, you can edit the lines in any order. (VS BASIC programs are kept in sequential files.)

Direct Files: are placed in the file in line number order. The line numbers must increment by one (that is, line numbers must be 1, 2, 3, 4, 5, etc.). All records must be the same length. Using VSPC commands, you can edit the lines in any order.

Object Program Files: are produced by the foreground processors. Such files consist of executable machine instructions, for example, compiled VS BASIC statements. You cannot edit these programs or use them for any other purpose.

Undefined Files: are special data files created during program execution. Only an object program can process or update such files, which can be either sequential or direct.

External: in addition, external VSAM files (that is, VSAM files outside of the VSPC library) are available for your use through the VS BASIC record-oriented I/O statements.

Filenames

When you save new work in a library, you become the *owner* of the file you create, and you must always specify a unique *filename* for it. When you specify a filename, you specify the following components:

libnum name/password

- *libnum*—Optionally, the *library number* (which is always the one-digit through seven-digit user number of the library manager).

  When you're referring to a file in your own library, you can omit the library number.

- *name*—Which is required, and which must be a unique identifying name within this library; a *name* is one through eight characters long—the first character being alphabetic, and the following characters being A through Z, 0 through 9, or $, #, or @.

- */password*—Optionally, the *file password.* It is described in the "File Passwords" section.)

Every file in your library has an *attribute* that you yourself control. The file attribute is the attribute your workspace had when you placed the file in the library. Because you yourself can change your workspace attribute, the file attribute is not necessarily the default attribute for your workspace.

No separator is required between *libnum* and *name;* however, one or more may be specified. No separator is required between *name* and */password;* however, one or more may be specified.

You must specify *libnum,* the library number, whenever you access any library except your own. (For your *project* library, you can specify *libnum* as 0 (zero).) You can omit *libnum* when you access your own (private, project, or public) library.

You must always include the name when you access any file.

When you access any file that was assigned a password when it was created, you must include the correct matching password immediately preceded by a slash (/); if you do not supply the password, you will be prompted to do so.

Special File CONTINUE

Every VSPC library can contain a special file with the filename CONTINUE. This file is used by VSPC to ensure that current work in your workspace is not accidentally lost.

The CONTINUE file is filled with the contents of your workspace in either of two ways:

- You issue an OFF CONTINUE command at the end of your session.
- The session is ended by conditions beyond your control (called a "forced ending").

Sometimes conditions make it impossible for VSPC to save your work. In this case a message tells you:

```
UNABLE TO SAVE 'CONTINUE'
```

When your work is saved, you can gain access to it at your next session with the command:

```
LOAD CONTINUE
```

If the file in your workspace was password-protected, you must supply the password to your LOAD CONTINUE command or you will be prompted for it.

CONTINUE usage conforms to the following rules:

- The user can retrieve a CONTINUE file only from his own library.
- Unless the user specifies OFF CONTINUE, a CONTINUE file is automatically removed from the library when the next OFF command is executed.
- A CONTINUE file can be overlaid by a CONTINUE file of another type.
- The SAVE command can specify CONTINUE. Note, however, that this destroys data already present in the CONTINUE file—data saved at the end of a previous terminal session.
- The FILE, PROTECT, SHARE, and STORE commands cannot specify CONTINUE.
- The CONTINUE file cannot be referenced by the input/output statements in a VS BASIC program.

Passwords

A *password* is a unique group of characters that you can use to limit access to VSPC itself or to your files. There are two kinds of passwords—*logon passwords* and *file passwords*. Both kinds of passwords have the same rules of formation: they can be from one through eight characters in length, they can be made up of any combination of alphabetic characters and numerals 0 through 9. You can specify passwords in lowercase letters, they'll always be translated to uppercase. For example, if you specify

```
TABOO02
```

or

```
taboo02
```

as a password, VSPC treats it as TABOO02. Note that the letter O and the number 0 are recognized as separate characters.

VSPC never displays your password at your terminal.

If you press ENTER when VSPC prompts for a matching password, VSPC responds as if all reprompting sequences have been executed. If you supply a password when none is required, VSPC treats it as an incorrect password.

If you press ENTER when VSPC prompts for a new or changed password, the password requirement is removed.

File Passwords

You can specify *file passwords* when you are saving files for future reference. You add the password at the end of the *filename.* Any time you retrieve the file, you must first supply the correct password before VSPC lets you access it.

A password can optionally be specified for each file in your library. These passwords can be created with the PROTECT, SAVE, STORE, or NAME commands. For example,

```
name file1/password
```

You can enter the command as shown above or you can, for additional security, enter the name followed by a slash (/).

```
name file1/
```

VSPC responds:

```
ENTER NEW PASSWORD FOR FILE1
```

For security purposes, some terminals suppress the display of passwords; others print a series of overstruck characters called a 'blot' over which you type your password.

When you attempt to retrieve a file protected with a password and do not supply it, VSPC prompts you to enter the correct password. If you do not supply the correct password after four attempts, the command is rejected.

Logon Passwords

The VSPC administrator can assign you a *logon password* in your user profile. You must provide this password before you are allowed to log on to work with VSPC. Once you have been given a *logon password,* you can change it during a terminal session. The next time you log on, you'll be required to provide the new password, not the original one.

Using the PASSWORD command you can change or delete your logon password.

To change your password, issue the PASSWORD command. You may include the new password as an operand of the PASSWORD command, or you may enter PASSWORD and wait for VSPC to prompt for the new password. When the ENTER PASSWORD message appears, enter your new password.

To delete your password, issue the PASSWORD command with no operand and, when the ENTER PASSWORD message appears, press ENTER. Your logon password is removed from your profile and, during subsequent logons, no password will be needed.

Remote Job Entry

The job entry facilities of VSPC provide you with a batch job entry and retrieval capability. The job control language and data for a job may be entered at the terminal and saved in your library. The job is submitted, upon your command, for batch processing. When the job is submitted, you may specify that printable output is to be made available for listing at the terminal. Appendix C contains a brief discussion of batch processing. For further information on VSPC batch facilities through job entry commands, see *VS Personal Computing (VSPC) General User's Guide and CommandLanguage.*

VSPC TERMINALS

Terminals, installed at remote locations, are connected to VSPC by telephone or leased communication lines. Different types of terminals supported by VSPC provide various features to meet the needs of a wide range of users.

The terminals you can use with VSPC are described in the following paragraphs. For each terminal there are several different keyboards available, and for each some of the features may vary. The variations are described in the *VS Personal Computing (VSPC) Terminals* manual.

IBM 3270 Information Display System

The IBM 3270 Information Display System can be one of several display terminals—each has a display screen, a typewriter-like keyboard, and other auxiliary equipment. Optionally, a printer and/or a light pen may be attached to the terminal. Several different keyboards are available for the variety of 3270 systems. One of the keyboard layouts for the 3275/3277 is shown in Figure 1; one for the 3276/3278 is shown in Figure 2.

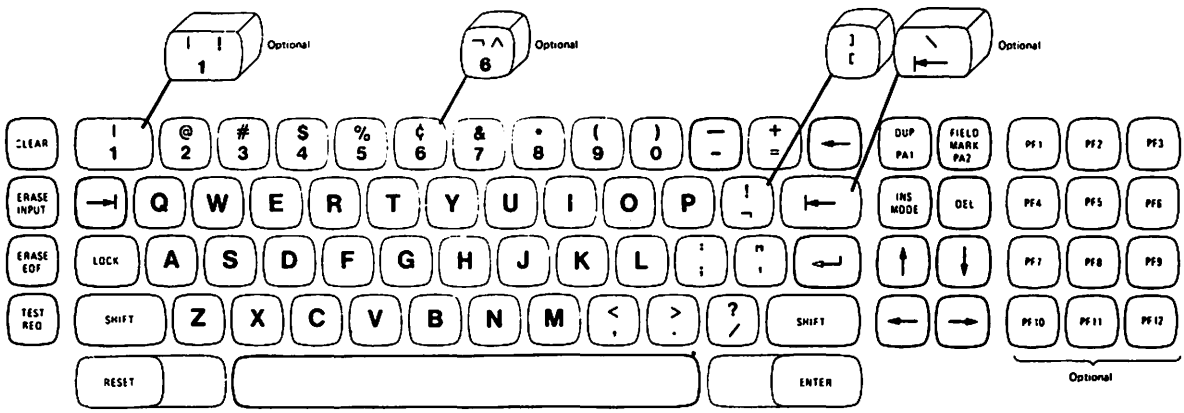

Figure 1. IBM 3275 and 3277 Typewriter-Like Keyboard

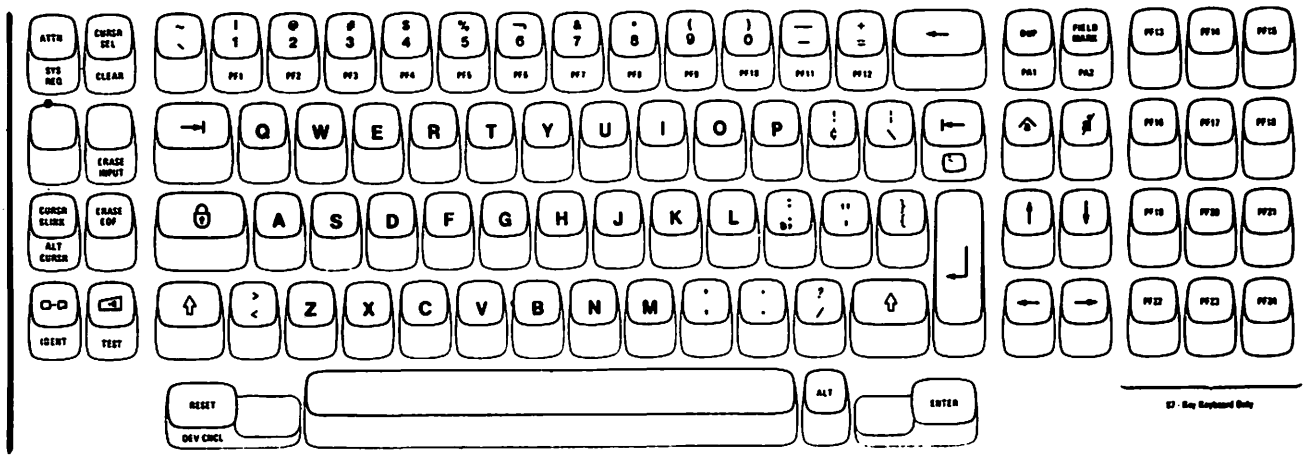

Figure 2. IBM 3276 and 3278 Typewriter-Like Keyboard

IBM 2741 Communication Terminal

The IBM 2741 Communication Terminal looks like and acts like a conventional IBM Selectric ® typewriter with two added controls that allow computer connection. One of the 2741 terminal keyboards is shown in Figure 3.

IBM 3767 Communication Terminal

The IBM 3767 Communication Terminal is a convenient desk top terminal that has a typewriter-like keyboard. When you are entering input, the print mechanism prints only in a forward direction; when it is printing computer output, however, the mechanism prints alternately forward and backward. There is an audible alarm that warns you when you are approaching the end of an input line or when system problems occur. A three-character indicator shows the next print position in the input line, or (when an error occurs) the cause of an error condition. Some controls on the 3767 vary, depending on

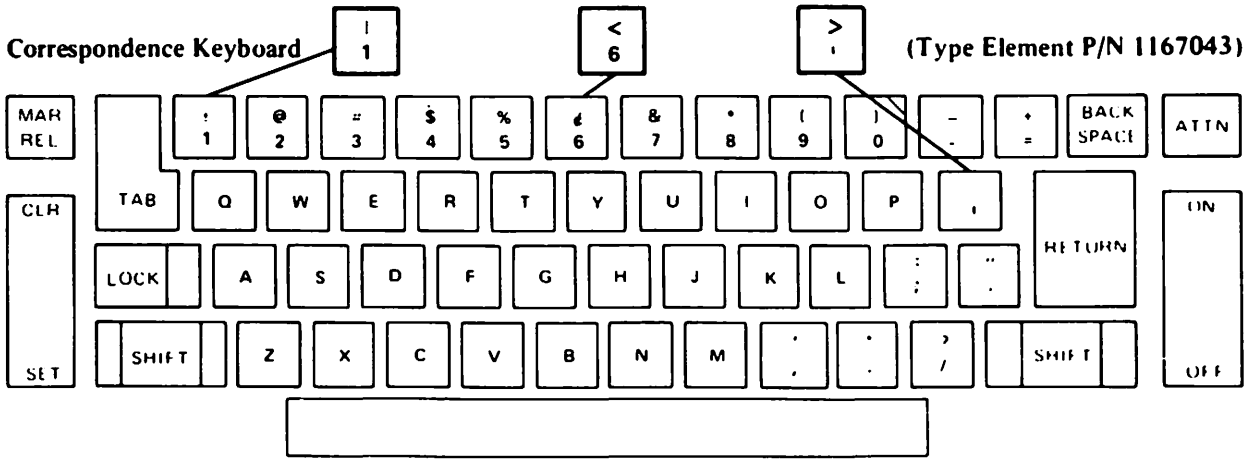

Figure 3. IBM 2741 Communication Terminal Correspondence Keyboard

whether you are in start/stop (S/S) mode or in synchronous data link control (SDLC) mode; your VSPC administrator can tell you which modes you can use. One 3767 keyboard is shown in Figure 4.

A typical layout of the 3767 terminal controls is shown in Figure 5.

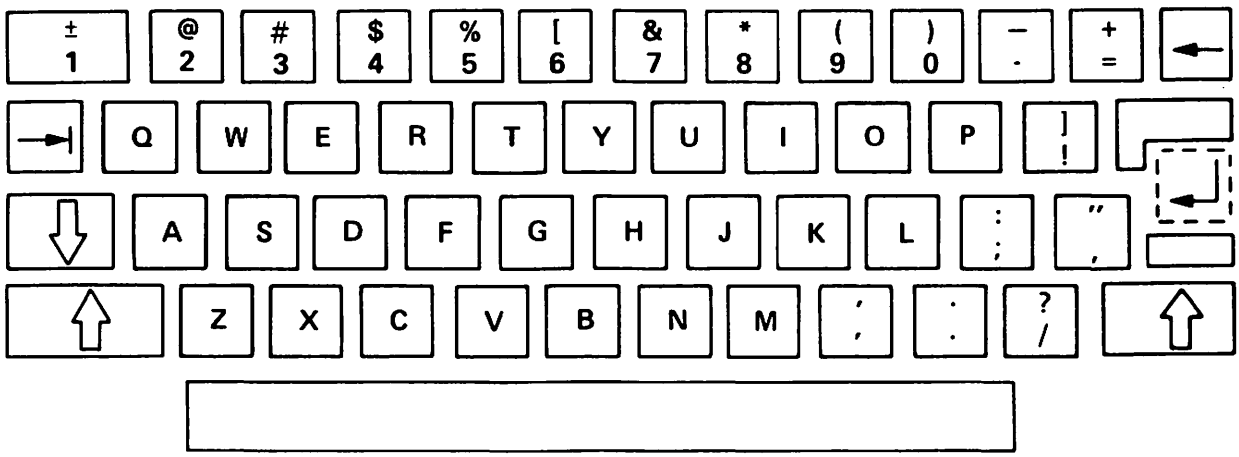

Figure 4. IBM 3767 Communication Terminal and Correspondence Keyboard

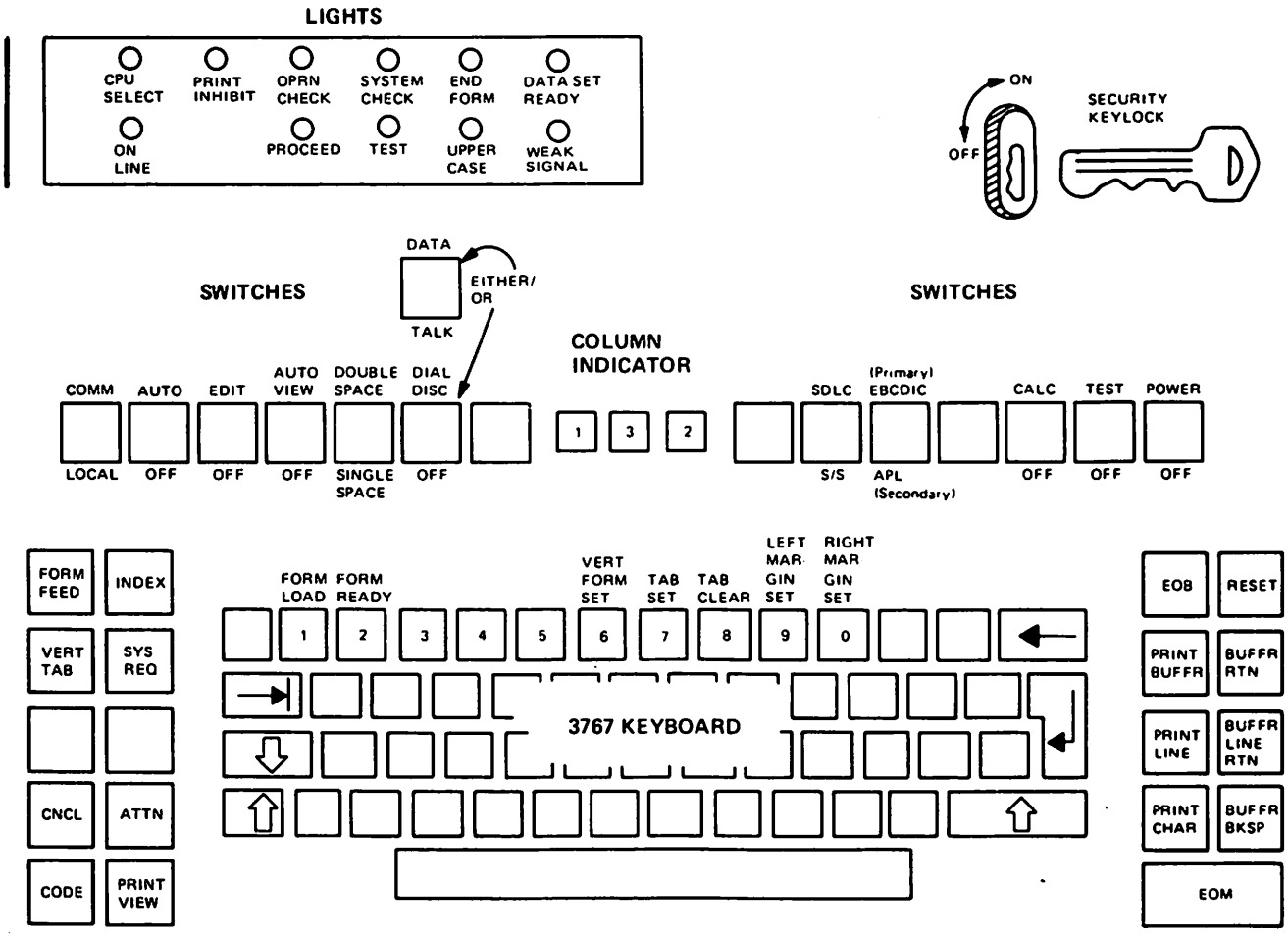

Figure 5. IBM 3767 Keyboard, Switches, and Light Layout

IBM 3770 Data Communication System

The IBM 3770 Data Communication System is a family of multipurpose keyboard/printer terminals. The IBM 3771 and 3773 terminals are designed for data entry, inquiry, and remote printing. The IBM 3774 and the IBM 3775 terminals each combine the features of the 3771 and 3773. For details on these terminals, see *VS Personal Computing (VSPC) Terminals.* For switch settings and logon instructions for the 3770, see the terminal reference chart in Appendix D of this book.

IBM 1050 Data Communication System

The IBM 1050 Data Communication System has special keys for computer connection plus a printer-keyboard that resembles a conventional IBM Selectric ® typewriter.

CPT-TWX (Model 33/35)

The CPT-TWX (Model 33/35) terminals include a control panel, a print mechanism, and a keyboard. All alphabetic characters are capital letters; special symbols act as control characters. The control panel is used to edit and transmit lines.

Note: The valid VSPC switch settings for each of the preceding terminals are shown in Appendix D.

VSPC TERMINAL-ORIENTED COMMANDS

The following activities are accomplished in the same way on any VSPC terminal:

- Specifying an attribute or password
- Formatting
- Displaying the status of your workspace, profile, or library
- Sending and receiving messages
- Signing off

Specifying a Workspace Attribute

If the workspace attribute specified in the logon message is the one you want to use, you can begin making entries. Otherwise, with the ENTER command, you can specify a different attribute.

ENTER Command

At the time your user profile was established, your VSPC administrator specified the default attribute your workspace assumes. This is the attribute it has every time you log on. You can explicitly change the attribute through the ENTER command.

ENTER	*processor-name* **CLIST** **DATA**	Required; choose one.

processor-name
specifies a workspace attribute associated with one of the IBM Program Products that run under VSPC, or a user-written compiler or interpreter so designed. The named processor must be installed with VSPC. Processor names for VS BASIC are:

BASIC (short precision)

LBASIC (long precision)

In addition to specifying precision in the ENTER command, precision can be specified in the OPTION statement of VS BASIC. Any precision specification in the OPTION statement overrides all precision specifications in the ENTER command.

CLIST
specifies a workspace that can contain VSPC command lists. (See "Command List (CLIST) Processing" under "Program Development.")

DATA
specifies a workspace attribute that is not associated with any specific processing capability.

If you specify BASIC (short precision), calculations will be represented to a maximum of seven digits. If you specify LBASIC (long precision), calculations will be represented to a maximum of 15 digits.

For example, if your program contains the following statements

```
120 let a = 141421356237309
130 print a
```

and you specify BASIC as a compiler option, A is printed as

```
1.414214E+14
```

If you specify LBASIC, you obtain greater precision, and A is printed as

```
141421356237309
```

Examples:

```
enter lbasic
```

gives you access to VS BASIC with long precision.

```
ent bas
```

gives you access to VS BASIC with short precision.

Adding or Changing Your Password

You can change an existing logon or file password or you can add a password if none exists.

PASSWORD Command

The PASSWORD command creates or changes your logon password.

PASSWORD	*password*	Optional; specify or omit. If omitted, password prompt is issued.

password
specifies your new logon password.

Your password must comply with VSPC rules for passwords. See "Passwords" under "Description of VSPC."

The new logon password replaces the one in your profile and will be required at your next logon.

For additional security, the PASSWORD command can be entered without specifying the replacement *password.* VSPC then responds with the prompt, ENTER PASSWORD, and the print suppression method or eight blots to disguise your password.

Once you are successfully logged on, you can delete your password. Issue the PASSWORD command and press carrier return instead of a password after the prompt and you will no longer be protected by a logon password.

If your profile does not specify a password, you can use the PASSWORD command to create one.

Examples:

```
password pass
```

creates a logon password of PASS.

```
pas
ENTER PASSWORD
```

In the last example, when you enter the PASSWORD command (pas) immediately followed by ENTER (or carrier return), VSPC prompts you (ENTER PASSWORD) and uses either the print suppression method or the blot for you to add a new password or delete an existing one by pressing ENTER (or carrier return) again.

Password Prompting

VSPC prompts for a password by displaying one of the following messages:

```
ENTER PASSWORD
ENTER PASSWORD FOR filename
ENTER NEW PASSWORD FOR filename
```

You enter the password.

During logon, ENTER PASSWORD is always displayed if a logon password is included in your profile.

In other commands, when the required file password is not supplied in the appropriate operand, a prompting message is displayed. It is issued when a file password is being created or changed or when a specific matching file password must be supplied.

Before the command can be executed, the valid password must be entered from the terminal.

If an incorrect matching password is entered, VSPC issues the following message:

```
INCORRECT PASSWORD - RE-ENTER
```

If the valid password is not supplied within three reprompting attempts, VSPC prints the following message:

```
PASSWORD ERROR
```

The command is then rejected.

If a syntactically invalid new or changed password is entered, only one attempt is allowed, and VSPC issues the following message:

```
PASSWORD ERROR
```

The command is then rejected.

Current Line Pointer

One of the lines in your workspace is considered your current line. (It is highlighted on the 3270 display screen in VIEW mode.) VSPC maintains a pointer to this line. Your current line pointer changes with the commands you issue during your editing session. Commands that use line numbers act on your current line when you enter an asterisk (*) in place of a line number.

As you enter data into your workspace (using the INPUT command or line-numbered entry), the pointer points to the last line entered. The pointer changes with several VSPC commands (for example, COPY, MOVE, and DELETE). You can change the pointer with the LOCATE command.

When you begin an editing session, the current line pointer points to the first line in your workspace. As you issue commands referring to different lines, the current line pointer is changed to the line you're working on.

When you're editing, there will be times you want to find the current line (it may not be displayed on your screen). You can use the LOCATE command to display or relocate your current line.

LOCATE Command

The LOCATE command finds or relocates the current line pointer. It can be relocated by absolute line number, by its display line position relative to the current line, or by its context.

LOCATE	*line* *+number* *−number* *begin-line* **'***string* **'** *begin-line:end-line* **'***string* **'** **'***string* **'** *	Optional first operand; specify one or omit. If omitted, current line is assumed.
	NOTEXT **TEXT**	Optional second operand; ignored in VIEW mode; otherwise, if omitted, NOTEXT is assumed.

line
specifies an absolute line number to which the current line pointer is to be set.

+number or *−number*
specifies a relative number of lines. The current line pointer is set higher or lower by the specified number of lines.

begin-line **'***string* **'**
specifies that the search for **'***string* **'** begin with *begin-line* and search to the end of the document.

begin-line:end-line **'***string* **'**
specifies a range of absolute line numbers to be searched for **'***string* **'**.

If *end-line* is omitted, the search begins at *begin-line* and goes to the end of the workspace. (The search does *not* then go to the beginning of the workspace and continue to *begin-line*.)

If both *begin-line* and *end-line* are omitted, the search begins with the line after the current line and continues to the end of the workspace.

'*string* **'**
specifies a set of characters for which VSPC is to search. The current line pointer is then set to the first line found that contains those characters.

*
specifies your current line. It can be used to specify *line*, *begin-line*, or *end-line* as defined above.

NOTEXT
specifies that only the line number of the new current line be displayed.

TEXT
specifies that the text of the new current line be displayed at the terminal after the line number.

When the LOCATE command is entered in VIEW mode, both TEXT and NOTEXT are ignored. In this case no explicit response is made, but the workspace display area is changed to reflect the new current line. In VIEW mode the current line is displayed and highlighted as the first line on the screen.

If an absolute line number is specified that does not exist, the current line pointer is set to the next lower line number. (If no lower line number exists, the current line pointer is set to the first line of the workspace.)

If LOCATE * is specified or if LOCATE is specified without a first operand, the current line pointer is found and VSPC displays the line number (if NOTEXT) or the line number *and* the text of the line (if TEXT).

Examples:

To find your current line:

```
locate
00050
```

or

```
locate * text
00010 BEGIN PROGRAM
```

The response tells you that 00010 is your current line. In VIEW mode this changes the display so that the current line is the first line (highlighted) at the top of the screen.

If you want to search for a certain set of characters:

```
locate 'x + y =z' text
00030 X + Y = Z
```

If *'string '* is specified and cannot be found, the current line pointer is unchanged and a message is issued:

```
LINE NOT FOUND
```

If the request is for a relative line number that exceeds the limits of the workspace, the current line pointer is set to the first line in the workspace if a negative relative line number is specified; or the last line if a positive relative line number is specified.

Formatting

Three VSPC formatting commands are useful to VS BASIC users.

- TABSET sets session tabs.
- TRANSLATE tells VSPC how to interpret each keystroke of your input and what to do with output.
- LINESIZE specifies the maximum length of the line to be displayed at your terminal.

Note: VSPC has a logical character-setting feature which is used to format data. When using VS BASIC, however, you should not attempt the use of the NEWLINE or BACKSPACE VSPC commands or the additional features of the TABSET command unless you understand completely how to use them. Consult *VS Personal Computing (VSPC) General User's Guide and Command Language* for details on these additional features.

TABSET Command

The TABSET command communicates session tab positions to VSPC.

TABSET	*tablist* **OFF**	Required; choose one.

tablist
specifies where session tabs are to be set. The tab list must consist of 1 through 26 integers representing tab positions relative to the left margin. The left margin is position 1. The integers must be written in ascending order with separators between them.

OFF
removes all current tab settings.

If you use tabs in your input, you must set them at each session because VSPC does not carry them over from one session to the next (assumes TABSET OFF at logon). Tab settings are not carried with your files. If you attempt to enter tabbed input without setting the session tabs, VSPC will not accept the line. On the IBM 3767 or 3770 terminals you do not need to manually set tabs; VSPC does it for you.

Tab positions can be changed during a session, but session settings should match the physical terminal settings. If they don't match, the appearance of your output may not match your input. You can display your current tab settings with the QUERY command.

When you have set session tabs (even if you don't use them in your input), VSPC uses them for output in combination with spaces and backspaces to speed up the printing. This use of tabs does not change the appearance of the material.

Examples:

```
tabset 5 10 15 20 25 30 35
```

sets session tabs at 5 10 15 20 25 30 35

```
tab off
```

clears session tabs.

TRANSLATE Command

The TRANSLATE command allows you to substitute a different set of characters for the ones on your keyboard.

TRANSLATE	*table-name* **CAPS** **OFF**	Required; choose one.

table-name
specifies the name of a predefined translate table. Ask your system administrator whether or not one is available.

CAPS
specifies that all alphabetic characters in your input be interpreted by VSPC as uppercase.

OFF
specifies that terminal input be accepted as entered.

Passwords and commands are unaffected by the TRANSLATE command (except for character strings in the CHANGE, FIND, and LOCATE commands).

When you sign on to VSPC, TRANSLATE CAPS is assumed and your input is interpreted as being entered as all capital letters.

Your system administrator may develop translate tables for use with VS BASIC. Ask if one is needed.

You can find out what translate table or other TRANSLATE operand is currently in effect with the QUERY command.

Examples:

```
translate mesa
```

translates your input to a predefined table named MESA (assuming there is one).

```
tra cap
```

translates your input to all uppercase letters.

LINESIZE Command

The LINESIZE command specifies the length of your output print line—from the left to the right margin.

LINESIZE	*n*	Required first operand.
	x	Optional second operand (CPT-TWX only); specify or omit. If omitted, previous specification unchanged.

n
specifies the number of print positions for an output line; it must be an integer from 18 through 255 (except with paper tape facilities on which the maximum value is 32,767, to permit output of large records to the tape). The left margin is print position 1.

x
specifies the number of idle characters to be used following each carrier return on a CPT-TWX terminal. It must be between 0 and 100. Idle characters may be necessary to ensure enough time for an actual carrier return on the CPT-TWX terminal.

The LINESIZE command does not affect the physical settings on your machine, but the assumed length in the computer. If you type beyond this length, VSPC accepts the line just as you've typed it; when VSPC sends the line back to you, the excess characters are displayed on the next line. If you set the value of n larger than the physical line length of your terminal, part of your terminal output may not be legible.

The value set by VSPC if you don't specify a linesize depends on your terminal:

IBM 3270 = 80
IBM 2741 = 120
IBM 3767 = 132 SLDC Mode
IBM 3767 = 120 S/S Mode
IBM 3770 = 132
IBM 1050 = 120
CPT-TWX = 72

Examples:

```
linesize 70
```

sets line length at 70.

```
lin 65
```

sets length of line at 65.

Questioning Your Status

Certain information is available to you about such things as your tab settings, the current name and size of your workspace, your workspace attribute, your user number, type of library access, the length of time you've been connected to the system, and which translate table you are using.

QUERY Command

This command displays information about your status in VSPC, characteristics of a file, and a list of files in a library.

QUERY	**WORKSPACE** **TABSET** **PROFILE** **ACCOUNT** **TRANSLATE** **FILE(***filename***)** **LIBRARY** **LIBRARY(***libnum***)**	Optional; choose one or omit. If omitted, WORKSPACE is assumed. (If libnum is omitted, user's own library is assumed.)
	OWN	Optional operand after LIBRARY or LIBRARY(libnum) only; specify or omit. If omitted, files of all owners are assumed.
	CONTENT(CLIST) **CONTENT(DATA)** **CONTENT** **(***processor-name***)**	Optional operand after LIBRARY or LIBRARY(libnum) only; choose one or omit. If omitted, names of all attributes are assumed.
	TYPE(SEQUENTIAL) **TYPE(DIRECT)** **TYPE(OBJECT)** **TYPE(SEQUENTIAL UNDEFINED)** **TYPE(DIRECT UNDEFINED)**	Optional operand after LIBRARY or LIBRARY(libnum) only; choose one or omit. If omitted, all file types are assumed.
	FROM(*name***)**	Optional operand after LIBRARY or LIBRARY(libnum) only; specify or omit. If omitted, first name is assumed.

WORKSPACE
lists current information about your workspace. This is the default.

Example:

```
query workspace

NAME:  RANDOM
CONTENT:  BASIC
LENGTH:  422 CHARS, 19 LINES
HIGHEST LINE NUMBER:   190
```

The series of messages tells you that your workspace currently has the name RANDOM, your current workspace attribute is BASIC, the workspace contains 422 characters in 19 lines, and the highest line number currently in use is 190.

TABSET
displays your current session tab settings or sends a message that you have set none.

Example:

```
query tabset

5 10 15 20 25 30 35 40 45 50 55 60 65
```

The tab list shows that you have set session tabs at character positions 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, and 65.

PROFILE
lists information about your user profile.

Example:

```
query profile

USER NUMBER:  1234567
LIBTYPE:  PRIVATE
CONTENT:  BASIC
CPUMAX:  5 SECONDS
SPACE:  2,000,000
SIZE:  20,000 EDITABLE,  60,000 OBJECT
INTPRMAX:  50,000 MAXIMUM,  20,000 DEFAULT
SSMAX:  4096 CHARS  10 ITEMS
JECODE:  DEPT27
PROJLIB:  456789
ACCOUNT:   '(A16729,707-A23)'JOB
```

This series of messages gives you information about your user profile, as follows:

USER NUMBER: 1234567—the user number you've been assigned.

LIBTYPE: PRIVATE—the type of library you have.

CONTENT: BASIC—your default workspace attribute.

CPUMAX: 5 SECONDS—the maximum CPU (central processing unit) time that can elapse between terminal inputs to a running program (to guard against loops). CPUMAX is omitted if it is zero (zero specifies the default option of unlimited time).

SPACE: 2,000,000—the maximum number of characters your files can occupy in any library.

SIZE: 20,000 EDITABLE, 60,000 OBJECT—the maximum number of characters in your workspace for editable data (20,000 EDITABLE) and for object programs (60,000 OBJECT).

INTPRMAX: 50,000 MAXIMUM, 20,000 DEFAULT—the maximum size (50,000 MAXIMUM) and the default size (20,000 DEFAULT) for the number of characters in an interpreter workspace.

SSMAX: 4096 CHARS 10 ITEMS—the maximum size (4096 CHARS) for a single data item in shared storage and the number of data items (10 ITEMS) you can make known to the shared storage manager at one time.

JECODE: DEPT27—your job entry code. (This entry also specifies that you are allowed to use the job entry facilities when they're available.)

PROJLIB: 456789—the number of your project library.

ACCOUNT: '(A16729,707-A23))' JOB—the accounting number your installation has assigned to you. If the word JOB appears, then this number is inserted as the accounting information in the JOB statement of any job submitted by you with a SUBMIT command.

The examples in this book assume this profile.

ACCOUNT
lists accounting information about your VSPC usage either as separate items or as composite processing units (or both), depending on how your installation has defined the accounting procedure.

TRANSLATE
identifies your current translation convention—*table-name,* CAPS, or OFF.

FILE(*filename***)**
specifies the selected file; *filename* must conform to the rules for a VSPC filename (see "Filenames" section). The file password, if any, is not required; however, if it is specified, it is checked.

When this form of the QUERY command is executed, the following information is displayed:

OWNER: *usernum* or OWNER: *usernum* RELEASED—The *usernum* of the file owner; only if this file resides in a project or public library. If RELEASED appears, indicates that the file is in a noncontrolled project library and has been released by its owner. See "ACQUIRE Command"and "RELEASE Command."

FILETYPE: *type* CONTENT: *content*—*Type* can be SEQUENTIAL, DIRECT, OBJECT, UNDEFINED SEQUENTIAL, or UNDEFINED DIRECT; *content* can be CLIST, DATA, or a processor-name (for example, BASIC or LBASIC).

FILE SIZE: *m* LIMIT: *n*—*m* is the present file size expressed in characters, and *n* is the maximum allowable size expressed in characters.

NUMBER OF RECORDS: *n*—*n* is the number of logical records in the file. (Omitted if the file type is OBJECT.)

RECORD SIZE: *n*—*n* is the length of the logical records in the file (displayed only if the file has the characteristics of a DIRECT or UNDEFINED DIRECT file).

SEQUENCE NUMBERS: *option*—*option* indicates the position and number of characters occupied by sequence numbers in each record when the file is submitted for batch processing outside the VSPC environment. The option can be specified by a NUMBER command, and the following conventions are used:

Option	Position
LIST *n*	left margin, followed by a blank
LEFT *n*	left margin
RIGHT *n*	right margin

In each case, *n* is the number of characters reserved for the number.

(If the NUMBER command specification for the file is NONE, this information is omitted.)

WORKSPACE LENGTH: *n*—*n* is the number of characters in an object program when it is running or loaded in main storage (displayed only if the file type is OBJECT).

ACCESS: *access-option*—*access-option* can be one or more of the options NOREAD, NOWRITE, or SHARE as specified by a previous PROTECT or SHARE command. (If the file has none of these access-options, the information is omitted.)

DATE LAST WRITTEN: *date*—*date* is when this file was last opened for output or update or a FILE, SAVE, or STORE command was issued for it.

DATE LAST READ: *date*—*date* is when this file was last opened for input or update, or a LOAD or RUN command was issued for it.

Note: The date can be in the form month/day/year or day/month/year, depending on the startup option chosen by your system administrator.

Example:

```
query file (1234567 file2)
or  q f (1234567 file2)
```

VSPC responds:

```
FILE TYPE: SEQUENTIAL CONTENT: BASIC
FILE SIZE: 7556 LIMIT: 8000
NUMBER OF RECORDS: 133
SEQUENCE NUMBERS: LIST
ACCESS: NOREAD
DATE LAST WRITTEN: 1/28/76
DATE LAST READ: 1/29/76
```

LIBRARY
lists names of files in your library.

LIBRARY *libnum*
identifies the library to be listed. If *libnum* is omitted, your own library is assumed. When *libnum* is specified, it must identify a library to which you normally have access. If a *libnum* of zero (0) is specified, your project library is assumed.

OWN (used only with LIBRARY or LIBRARY(*libnum*))
indicates that only files owned by the requestor within the specified project or public library are to be displayed.

When OWN is specified for a project or public library, only your own files are included in the display. OWN has no meaning when specified for a private library.

CONTENT (used only with LIBRARY or LIBRARY(*libnum*))
specifies that files with the parenthesized attribute are to be displayed.

When CONTENT is specified, only names of files with the specified attribute (CLIST, DATA, or *processor-name*) are displayed. The *processor-name,* when specified, must be valid for this VSPC installation. It may be an IBM program product processor (for example, BASIC or LBASIC), or a user-written compiler or interpreter.

TYPE (used only with LIBRARY or LIBRARY(*libnum*))
specifies that only files of the parenthesized type are to be displayed.

When TYPE is specified, only files containing files of the specified type (SEQUENTIAL, DIRECT, OBJECT, UNDEFINED SEQUENTIAL, or UNDEFINED DIRECT) are included in the display. Note that UNDEFINED, UNDEFINED SEQUENTIAL, and SEQUENTIAL UNDEFINED can all be specified and have identical meanings, and that UNDEFINED DIRECT and DIRECT UNDEFINED can both be specified and have identical meanings.

FROM (used only with LIBRARY or LIBRARY(*libnum*))
specifies the name of the file at which listing is to begin. Displaying of the list begins with the first *name* that is equal to or higher than the specified *name,* and continues through the end of the library in ascending order as follows: $,#,@,A,...,Z,0,...,9.

If you signal attention during the display, the display will cease.

If no options are specified, a list of all names in your library is displayed.

Examples:

```
q lib(456789) content(basic) from (RANDOM)
```

displays a list of the names of all short precision VS BASIC files in your project library, starting with RANDOM and continuing through the rest of the library.

```
query library
```

displays a list of all names in your own library.

Message Facilities

VSPC lets you send messages to other users, to the system operator, or to the VSPC operator. You can also control the receipt of messages sent to your terminal.

SEND Command

The SEND command transmits messages to other terminals.

SEND	*'message-text '*	Required first operand.
	usernum **CONSOLE** **OPERATOR**	Optional; choose one or omit. If omitted, message is sent to OPERATOR.

'message-text '
is the actual text of the message to be transmitted. It may contain any combination of the following characters:

the 26 letters A through Z
the digits 0 through 9
13 special characters / + - * . , = ( )
' ? : and ;

usernum
identifies a VSPC user as the recipient.

CONSOLE
identifies the operating system console as the recipient.

OPERATOR
identifies the VSPC operator as the recipient.

The text of your message can be as many as 64 characters long and must be enclosed in single quotation marks.

Alphabetic characters in messages are translated into uppercase even if TRANSLATE OFF is in effect.

If you want an apostrophe or a single quotation mark within your message, you must use two adjacent single quotation marks. These will appear as the one-character single quotation mark in the text of your message.

If you don't specify a destination, the VSPC operator receives your message.

If you send a message to a user who is not logged on, you are informed of this and your message is not delivered. If you send a message to a user who has specified MESSAGE BLOCK, your SEND command is rejected. If the message is not displayed at the user's terminal before he logs off, the message is purged.

Examples:

```
send 'how are you?' 649299
```

sends the message 'HOW ARE YOU?' to user number 649299.

```
sen 'what time is it?'
```

sends the message 'WHAT TIME IS IT?' to the VSPC operator.

MESSAGE Command

The MESSAGE command determines whether or not messages are displayed at your terminal and whether or not identifying headings are displayed with your messages.

MESSAGE	**ID** **NOID**	Optional; choose one or omit. If omitted, most recent specification remains in effect. If never specified, NOID is assumed.
	BLOCK **OPEN** **WAIT**	Optional; choose one or omit. If omitted, most recent specification remains in effect. If never specified, MESSAGE OPEN is assumed.

Message identification headers:

ID
specifies that identification headers be displayed with messages.

NOID
specifies the suppression of the display of identification headers with messages.

All messages are stored with a unique identification. However, since most VSPC messages are self-explanatory and relate directly to current activity, message identifications are not normally required.

If you specify MESSAGE without either of these operands, the current display convention remains in effect.

Message receiving mode:

BLOCK
prevents the display of messages sent to your terminal by the SEND command.

OPEN
returns the terminal to normal operation and allows the receipt of messages whenever the terminal is not open for input. Terminal input can be entered without a previous attention signal.

WAIT
puts the terminal in a continuous state of readiness to accept messages; you can't enter input until you signal attention.

If either BLOCK, OPEN, or WAIT is not specified, the current display convention remains in effect. But if ID and NOID are also omitted, MESSAGE OPEN is assumed.

Note: If MESSAGE BLOCK is changed to MESSAGE OPEN, the most recent general message from the VSPC operator is displayed (if any was sent after MESSAGE BLOCK was issued).

Examples:

```
mes wait
```

tells VSPC you want your messages displayed whenever they are sent; an attention signal must precede any input.

```
message id open
```

tells VSPC you want to receive messages with their identification whenever they are sent. A VSPC message received with the above command operands in effect will include an identification header (beginning with the letters ASU), for example,

```
ASU201 READY
```

A VS BASIC message received with the same command operands in effect also has an identification header (beginning with the letters ICD), for example,

```
ICD457 TOO LITTLE DATA..RETRY
```

The same messages with the following command operand in effect

```
mes noid
```

have no identification header:

```
READY
```

```
TOO LITTLE DATA..RETRY
```

Explanations of VSPC terminal messages are listed in a computer printout that can be obtained by the procedure described in the *VSPC General User's Guide and Command Language.*

Ending a Session

To end a terminal session, enter the OFF command.

OFF Commmand

The OFF command ends your terminal session.

OFF	**CONTINUE**	Optional; specify or omit. If omitted, workspace contents not saved in CONTINUE file.

CONTINUE
saves your workspace contents so that it will be available when you start your next session. If CONTINUE is not specified, anything that is in your CONTINUE file is erased when you sign off. If you haven't saved your workspace contents and do not specify CONTINUE with your OFF command, the contents of your workspace is lost.

If the file in your workspace has a password when you specify CONTINUE, you must use it when you load the CONTINUE file.

Examples:

```
off continue
```

terminates connection with VS BASIC and VSPC and puts the contents of your workspace in the file named CONTINUE.

```
off
```

terminates connection with VS BASIC and VSPC and erases the contents of your CONTINUE file, if you had one.

The system responds to the OFF command with some messages:

```
date time usernum
CONNECTED time
CPU TIME sss
```

- date time displays the date and time you logged off.
- usernum states your user identification number.
- CONNECTED indicates the elapsed time since you logged on.
- CPU TIME indicates the time in seconds used by the central processing unit (CPU) to complete your requests.

Forced Endings

A "forced ending" occurs when your session ends unexpectedly due to circumstances beyond your control. Your keyboard locks and your efforts to gain a system response fail. The effect of a forced ending is similar to your issuing the command OFF CONTINUE. In most instances your workspace is saved under the name CONTINUE.

CONDUCTING A 3270 TERMINAL SESSION

Conducting a session on a 3270 Display System differs from the method used for nondisplay terminals. But to use VSPC on any terminal, you must first contact the computer, then log on to VSPC before you can type data and commands and use the VSPC editing facilities.

Entering and editing data or commands in VSPC can be handled in two distinctly different ways on 3270 terminals.

When you log on to VSPC your entries (commands and data) appear at the bottom of the screen in the input area (see Figure 6). When you press ENTER, these lines appear in the output area. You can edit lines in the input area with the general editing procedures described in this section. In this book this screen format and method of input and editing is referred to as "command mode."

When you issue the VIEW command, your screen format changes to allow direct editing of lines in your workspace. This book refers to this screen format and method of input and editing as "VIEW mode.".

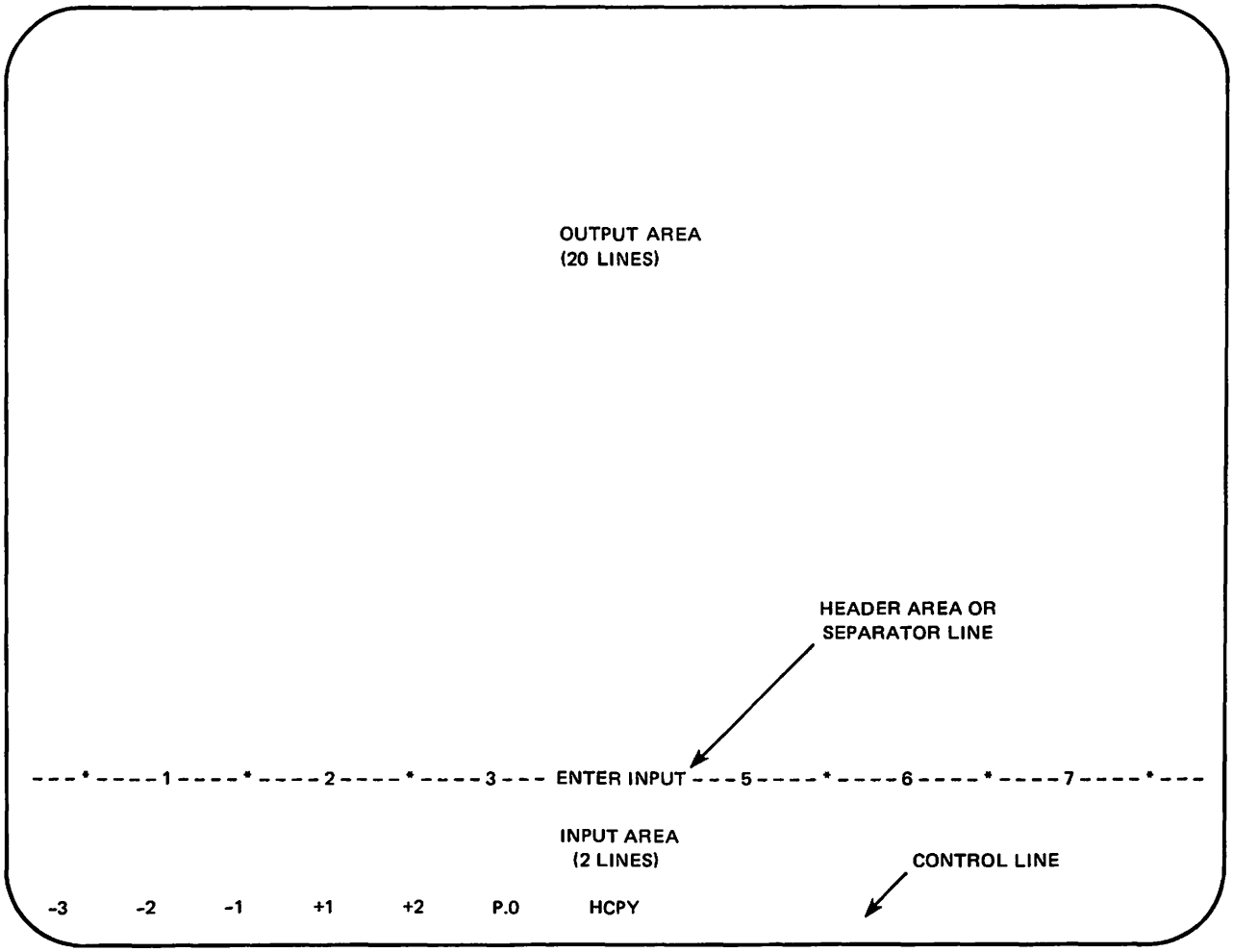

Figure 6. IBM 3270 VSPC Screen Format in Command Mode

Contacting the Computer

An IBM 3270 can be locally connected to a computer system through a channel, or remotely connected through a nondial (leased) line. These instructions assume a display station equipped with the IBM Line Adapter feature. If your stations is not so equipped, check with your VSPC administrator about how to establish a connection using the equipment at your location.

Pull out the switch labeled OFF-PUSH to apply power to your display station. The SYSTEM AVAILABLE light turns on and the cursor appears in the upper-left portion of the screen.

VPSC Command (Logon)

After establishing contact with the computer, press the CLEAR key and enter your logon command as follows (no abbreviation):

VSPC	**ID=***usernum*	

ID= *usernum*
identifies you to VSPC. This number is assigned by your VSPC administrator. A usernum contains from 1 to 7 digits; a value of zero (0) is not a valid user number.

Press the ENTER key.

Note: For certain 3270 Display Systems that use the SDLC line discipline the TEST REQ key may be required instead of ENTER. If your IBM 3277 or 3278 is connected via a 3791, the procedure is also different. Consult *VSPC Terminals* or check with your VSPC administrator.

If your logon is not successful, you will receive a message stating the reason. You then must repeate the logon command, correcting the indicated problem. If you receive no response after typing your logon request, VSPC is not available.

If your logon is successful, the VSPC screen format appears as in Figure 6. If your installation requires a password, VSPC prompts:

```
ENTER PASSWORD
```

When you type your password, it does not display. If the password you provide is not accepted, VSPC repeats the prompt and you can enter another password. If you have not provided the proper password within the number of prompts allowed (four) or within the allotted time (two minuts), you are disconnected from VSPC.

When you have successfuly logged on (and typed a valid password), the following messages are displayed:

```
attribute time date usernum
[CONTINUE attribute]PASSWORD]]
[logon-message text]
READY
```

- attribute designates the content classification of your workspace as specified in your profile.
- time data displays the time and date you logged on.
- usernum states your user identification number.
- CONTINUE (when present) indicates that the contents of your workspace at the end of your last terminal session have been saved in the CONTINUE file and are available again with a LOAD CONTINUE command.
- *attribute* (in the second line) is the attribute of your CONTINUE file.
- PASSWORD is printed only if CONTINUE is protected with a file password. When you load the CONTINUE workspace, you must supply the correct password.
- *logon-message-test* (when present) is a general information message which may change periodically.
- READY indicates VSPC is ready to accept input.

If the workspace attribute specified is the one you want to use, you can begin making entries. Otherwise, with the ENTER command, you can change your workspace attribute. See *VS Personal Computing (VSPC) General User's Guide and Command Language* for additional information.

Example of 3270 Logon

You contact VSPC:

```
vspc id=1234567 @
```

VSPC prompts for your password:

```
ENTER PASSWORD
```

(You enter your password, but it does not display.) After you type your password and press ENTER, VSPC sends you some messages:

```
BASIC  02/04/78  09:16:20  1234567
CONTINUE BASIC
WELCOME TO VSPC
READY
```

The READY response is standard after most successfully executed commands and indicates that VSPC is ready for your next entry. An explanatory error message is issued to your terminal if VSPC is unable to execute your command.

General Editing

The VSPC screen format shown in Figure 6 is in effect when you log on to VSPC. When you type at the keyboard, your data or commands appear on the screen in the input area on the lower part of the screen. As you enter more lines and as VSPC responds to you, the entire dialog appears in the output area. On the control line you specify terminal actions—either by use of an optional light pen or by cursor positioning. The header area tells you what kind of processing you are currently using.

With this method you tell VSPC what to do from your terminal, and VSPC responds. Responses from VSPC vary depending on the type of terminal and whether you are typing lines of data or running a program. This interaction may vary on some 3270 terminals; this is the general procedure.

1. VSPC notifies you in one or more of the following ways that it is ready to accept your input:
   - Lights the SYSTEM AVAILABLE indicator
   - Displays a question mark (?)
   - Displays a message (for example, READY)
   - Displays a line number (for example, 00010)
2. You type a line of input as follows:
   a. Type the line of data or a command.
   b. Press ENTER.
   c. Wait for VSPC to respond in one or more of the above ways before you type another line.

You can use all of the terminal editing facilities (INS MODE, DEL, etc.) to modify the lines displayed in the input area as described below.

Character Correction

Move the cursor back to the incorrect character with the cursor control key; then type the correct character.

Character Insertion

Press the INS MODE key and move the cursor to the position in which the additional character is to be inserted. Type the missing character. As characters are inserted, all characters to the right of the cursor are shifted to the right. If the insertion causes the line to exceed the line limit, the excess characters shift from the end of the first line of your input line to the beginning of the next. The INPUT INHIBITED indicator turns on and your keyboard is disabled if you try to insert more characters than the area will hold.

Press the RESET key to return the keyboard to its normal mode of operation and turn off the INSERT MODE indicator.

Character Deletion

Move the cursor to the position of the extra character and press the DEL key. The characters to the right of it move left one space.

Line Correction

With the backspace key, return the cursor to the first character of the data you want to replace. Then press the ERASE EOF key. This erases all characters from the cursor position to the end of the line. Retype the line.

Special Requests

Three terminal conditions allow you to make special requests.

1. When the MORE...PRESS CLEAR message is displayed on the screen, in addition to pressing CLEAR to continue output, you can:
   - Press PA1 to signal attention
   - Request backpaging
   - Request hardcopy
   - Press PA2 to cancel output
   - Press ENTER to hold the display longer than the automatic change time of 30 seconds. The message changes to HELD...PRESS CLEAR and any of the above actions may be taken before pressing CLEAR to continue output.
2. When ENTER INPUT appears on the screen, in addition to entering input, you can:
   - Press PA1 to escape from input
   - Request backpaging
   - Request hardcopy
3. When the terminal is idle, you can:
   - Press PA1 to signal attention
   - Request backpaging
   - Request hardcopy

Note: If the INPUT INHIBITED light is on, and you want to signal attention, you can press the RESET key and then press the PA1 key (which signals attention). You can continue this sequence until VSPC begins an action in response. Note also that some foreground processors may not allow attention signalling in this situation.

Display Screen Hold

When your output needs more than the remaining space available on the screen to display, VSPC pauses at the end of the screen with MORE...PRESS CLEAR shown in the screen separator line. Pressing CLEAR displays the next page.

If you press neither CLEAR nor ENTER within 30 seconds, the next page is displayed automatically.

If, while MORE...PRESS CLEAR is showing, you press the ENTER key, the message changes to HELD...PRESS CLEAR and the current display remains until you press the CLEAR key.

Invalid Output Characters

If you find a double quotation mark (") in your output that you didn't expect, VSPC has found an invalid character in your data.

Program Function (PF) Keys

The program function keys provide shortcuts to entering commands, sets of characters, or control functions. You can assign a command, a set of characters, or a control function to any PF key on your terminal with the PFKEY command. They are interpreted the same way in both screen formats with one exception: If the FORWARD or BACKWARD option of the PFKEY command is used in VIEW mode, the workspace display area is moved instead of normal paging.

PFKEY Command

The PFKEY command specifies the interpretation of the program function (PF) keys on the 3270 terminal.

PFKEY	*n*	Required first operand.
	' ' '*string* ' '*string* ' **ENTER** '*string* ' **NOENTER** **LAST** **LAST ENTER** **LAST NOENTER** **HCPY** **FORWARD**(*x*) **BACKWARD**(*x*)	Required second operand; choose one.

n
specifies the number of the PF key to which you want to assign a '*string* '. It must be between 1 and 24 inclusive.

' '
this operand (two adjacent single quotation marks) specifies that any currently specified PF key definition be removed.

'*string* '
specifies a string of characters to be inserted into the screen image at the location of the cursor when the selected PF key is pressed. This use of a PF key is the same as entering the characters after pressing 'INS MODE' on the terminal and typing a string of characters. The cursor is relocated at the character position following the inserted string.

The maximum length of a string is 255 characters. The maximum total length of character strings assigned to PF keys is 600 characters.

LAST
specifies that when the PF key is pressed, the contents of the last previous terminal input line into the general command area be inserted at the current location of the cursor. The cursor is repositioned following the inserted characters.

ENTER
specifies that the string be copied to the screen as described above, and then the appropriate action taken as if the ENTER key had been pressed.

NOENTER
specifies that the string be copied to the screen but no other action taken. You can then modify it before pressing ENTER. Your line is truncated if the insert extends beyond the current input field.

If neither ENTER nor NOENTER is specified, ENTER is assumed.

HCPY
specifies that subsequent use of the named PF key should simulate selecting the HCPY field on the screen through cursor positioning and the ENTER key.

FORWARD(*x*)
specifies that subsequent use of the named PF key should cause the display to be paged forward a specified number of pages. The number of pages may be any number between 1 and 99.

BACKWARD(*x*)
specifies that subsequent use of the named PF key should cause the display to be paged backward a specified number of pages. The number of pages may be any number between 1 and 99.

Examples:

```
pfkey 4 'goto 40'
```

sets PF key number 4 so that every time you press it, the character string 'goto 40' is inserted at the location of the cursor.

```
pfkey 3 forward (1)
```

pages your screen forward 1 screen every time you press PF 3.

VIEW Mode

When you issue the VIEW command, your screen is considered to be in VIEW mode, and the format shown in Figure 7 appears on your screen. Your active workspace is displayed with line numbers in the 20-line workspace display area. You can edit directly with the general editing procedures described above.

The separator line corresponds to the Header Area in Figure 6. It indicates whether you are in VIEW mode, INPUT mode, or another mode.

You enter VSPC commands in the general command area.

The control line contains a message area to display messages and responses to your commands. If a message is too long to fit, it is displayed in the workspace display area. When this happens, press CLEAR and your current screen is restored. You can point to the other items on the control line with the cursor, CURSR SEL key, or light pen.

- +1 or −1 moves your display one page forward or backward without changing the current line pointer.
- HCPY requests the current display be printed. (You must have selected a printer with the HARDCOPY command.)
- *xxxxx* displays your current line number. If you point to it with cursor, CURSR SEL, or light pen, your current line is moved to the top of the screen.

Note: If you use the light pen to select fields and cause a change in the display

screen, any direct entry or editing changes you've made since you last pressed ENTER are lost. Use PF keys or cursor positioning and ENTER, to be sure your changes are saved in your workspace.

VIEW Command

The VIEW command allows you to use the direct input/edit facilities of VSPC. After issuing this command, your screen is considered to be in VIEW mode and will change from that in Figure 6 to that in Figure 7. To get out of VIEW mode, press the ENTER key when you have made no changes to the screen. Certain VSPC commands (RUN, STORE, or ENTER) also take you out of VIEW mode (change your screen format back to that shown in Figure 6).

If you are already in VIEW mode and issue the VIEW command, a different part of the workspace is displayed without altering the current line pointer.

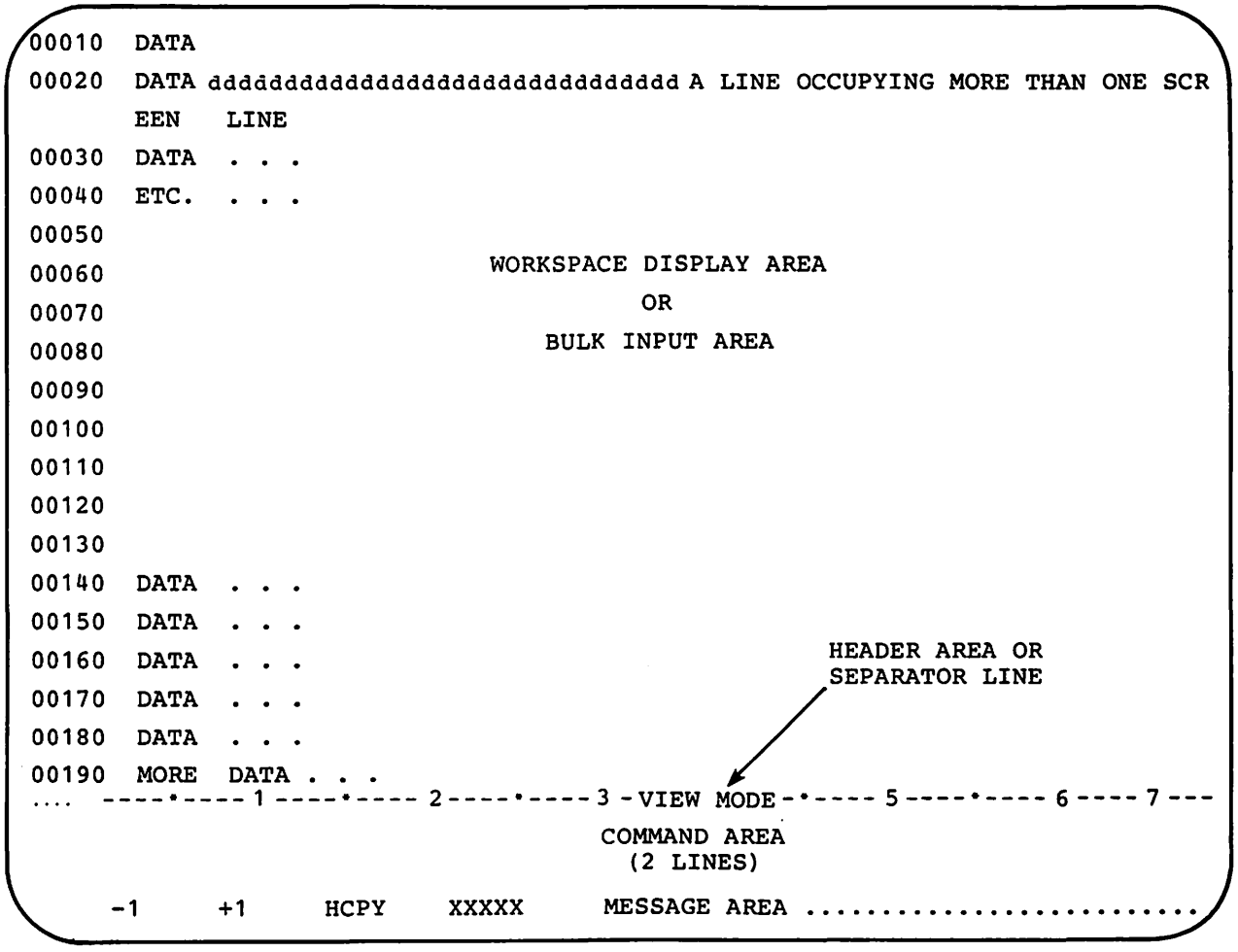

Figure 7. IBM 3270 VSPC Screen Format in VIEW Mode

VIEW	*line* * +*line* −*line*	Optional; choose one or omit. If omitted, current line is displayed at top of screen.

line
specifies an absolute line number. The number and the text of the line will be displayed at the top of the output area. The current line pointer is not changed.

*
specifies that your current line be displayed at the top of the screen.

+*line* or −*line*
specifies the relative number of lines the display is to be moved. +*line* moves the display toward the end of the workspace; −*line* toward the beginning.

If the VIEW command is issued with no operand during direct edit, the screen is displayed with the current line at the top of the workspace display area.

With direct edit, any command that changes the contents of the workspace (RENUMBER, MOVE, DELETE) updates the display. Any command that moves the current line pointer to a line not displayed on the screen redisplays the workspace with the new current line at the top of the screen.

Responses to commands you issue in VIEW mode are displayed on the control line if they fit. If not, your screen is taken out of VIEW mode and the VSPC screen format shown in Figure 6 is displayed with the response as the last line of the output area and MORE... PRESS CLEAR in the separator line. In this case, press the CLEAR key to return to VIEW mode.

Examples:

```
view
```

changes the screen format from that in Figure 6 to that in Figure 7. You are now in VIEW mode and can use the direct editing facilities.

```
view 30
```

puts your screen in VIEW mode and changes the current line pointer at the same time.

If you press ENTER with no changes on the screen, your screen is taken out of VIEW mode and you can page back to see the VSPC commands you've issued and their responses.

Local Commands

Local commands are used only in VIEW mode. They are entered over a line number displayed in the output area starting in the left margin of the screen. Any characters after the command are ignored. These commands usually specify an action to be taken on the line over whose number they are entered. The actual number of the line is normally not changed by the entry of the command, and is restored when the screen is redisplayed following execution of the command.

The form of a local command is:

```
nnnnX
```

nnnn
is any number of lines (up to four digits) you want to add, delete, or repeat.

X
is the command character: A, D, R, or /.

A—Add lines
The specified number of blank lines is added to the workspace after the selected line. The cursor is positioned at the start of the first of these lines. Line numbers for the added lines are generated in increments of one beginning with the line number of the selected line plus one. If there are insufficient line numbers available before the next existing line, lines are renumbered in increments of one as far as is necessary. If this occurs, a warning message is displayed in the message area of the control line. The added lines have a length of one character unless the workspace contains a 'DIRECT' file, in which case the length is the same as that of the selected line.

D—Delete lines
The specified number of lines is deleted from the workspace beginning with the selected line. The lines being deleted are deleted without inspection; any local commands entered on them are ignored.

R—Repeat lines
The specified number of lines is added after the selected line with identical text. Line numbers for the added lines are generated in the same manner as for the add lines command (A).

/—Change current line pointer
The selected line becomes the current line of the workspace. If this command is entered more than once on the same screen, the indicated line nearest the bottom of the screen becomes the current line.

You may enter multiple local commands on the screen before you press the ENTER or a PF key. The commands take effect in a sequence determined by their physical position on the screen, starting at the top.

Examples:

To make line 00050 instead of 00060 the current line, position the cursor under any digit of line 00050 and type the slash:

```
00040 data A
/0050 data B
00060 data C
```

Then, when you press ENTER, line 00050 becomes your current line.

```
00040 data A
00050 data B
00060 data C
```

You can add lines in the workspace:

```
00040 data A
4A050 data B
00060 data C
```

VSPC adds four blank lines after line 00050:

```
00040 data A
00050 data B
00051
00052
00053
00054
00060 data C
```

Nonlocal Keys

While in VIEW mode you can move your cursor around the screen, type over data, insert characters (INS MODE), and delete characters (DEL) to modify the display of your workspace.

The nonlocal keys (CLEAR, ENTER, and the Program Access (PA) keys) are interpreted in a special way in VIEW mode.

CLEAR Key

CLEAR nullifies changes made to the displayed text and displays the screen as it was before you made the changes.

Another use of CLEAR is to exit from 'message mode.' When a message or a response resulting from a command occupies multiple lines or is too long to be displayed on the message portion of the control line, enter 'message mode.' VSPC displays the command mode (Figure 6) screen with the message as the last line in the output area, and MORE ... PRESS CLEAR in the separator line. Press the CLEAR key to return to VIEW mode.

ENTER Key

Press the ENTER key to signal that all your changes are to be entered into the workspace. If ENTER is used with no modifications made to the display, you are returned to command mode.

Program Access (PA) Keys

These program access keys are ignored during direct edit. While in message mode, PA2 may be used to cancel output.

Order of Processing

When you press ENTER, a PF key, or make a selection on the control line, the changes and requests are processed in the following order:

- Selecting a field with the light pen or CURSR SEL:
  1. If you select HCPY, the screen image will be printed as it was the last time you pressed ENTER or a PF key. The changes you made will remain so that when you press ENTER the next time, your changes will be made.
  2. If you select +1, −1, or the current line number, the screen is immediately rewritten as requested. The prior changes you made are lost.
- Selecting a field with the cursor and the ENTER key:

2. A HCPY request causes the updated screen to be printed.
3. +1 or −1 displays the appropriate new page *after* applying any updates entered on the screen.

- Selecting a field by positioning the cursor before using a PF key:

  Changes on the screen are made before a HCPY or forward or backward paging request.

A PF key set to a character string with the ENTER option is processed as if the string were inserted at the location of the cursor and then the ENTER key pressed.

All commands are processed before the screen is redisplayed.

Cursor Positioning

After processing the modifications to the screen, the new screen is formatted for the next display. When this new screen is displayed, the cursor is positioned at the start of the command area unless any of the following apply:

- You used a program function key with the NOENTER option to insert a set of characters. In this case the cursor is positioned after the string associated with the PF key.
- You issued the 'A' or 'R' local command. In this case the cursor is positioned at the start of the added lines. However, if another command entered on the same screen causes the workspace display to change such that the lines are not displayed, the cursor is at the start of the command area.

  If you enter multiple 'A' and/or 'R' commands on the same screen, the cursor is set to the position corresponding to the command nearest the top of the screen.
- You issue the INPUT command. In this case the cursor is positioned at the start of the area for adding new text.

HARDCOPY Command

For IBM 3270 Information Display Systems, the HARDCOPY command specifies which printer you want to use.

HARDCOPY	*node-name* * **OFF**	Required;choose one.

node-name
must be predefined. This operand requests the use of a separate 3270 Display System printer.

*
indicates that a 3284 printer is directly associated to a 3275 display unit, and that the node-name for the display unit is also that for the printer.

OFF
specifies that the printer designated by the node-name currently in use is to be disconnected.

The HARDCOPY command must be executed before printing operations can be requested.

The * and *node-name* operands are used to request the use of a printer. If the indicated printer is available, VSPC responds with the following message:

```
READY
```

Requests for printing operations can then be entered.

Printing of the currently displayed page is requested by positioning the cursor at the HCPY field on the control line and pressing the ENTER or CURSR SEL key, by use of a program function (PF) key predefined as HCPY, or, if available, by selecting the HCPY field with the light pen. The current page is then printed.

If the indicated printer cannot be obtained, VSPC responds with the following message:

```
DEVICE NOT AVAILABLE
```

In this case, no request for printing can successfully be executed at this time.

The OFF operand is specified when the user wishes to release the previously obtained printer. When execution is successful, VSPC responds with the following message:

```
READY
```

The printer previously selected is no longer available for use.

Examples:

If your 3270 display terminal has an associated printer, you can request that printer services be made available to you through the HARDCOPY command.

```
hardcopy *
```

On the 3275 terminal, this command specifies that output is to be sent to the attached 3284 printer.

A 3270 Information Display System printer that is not physically attached to your 3270 display terminal, but is shared by several terminal users, may be associated with your display terminal by entering:

```
hardcopy node-name
```

where node-name is a name specified by your installation to identify the particular printer.

Later, when you want to print the current page, you can do so with the HCPY field in the control line.

When you no longer need the printer, specify

```
har off
```

This command disconnects the printer.

CONDUCTING A 2741 OR 3767 START-STOP TERMINAL SESSION

On any terminal you must first contact the computer, then log on to VSPC before you can type data and commands and use the VSPC editing facilities.

Contacting the Computer

The procedure you use to contact the computer varies depending on whether the connection is through a leased line or a switched line. Your VSPC administrator will tell you which type of connection you're using.

If your connection is by a leased line, you can enter the VSPC logon command as soon as you turn the power switch on.

If your connection is by a switched (telephone) line, then the additional steps shown in Figure 8 are necessary.

Telephone Data Set	Acoustic Coupler
1. Press TALK button.	1. Make sure coupler is: connected to power, turned off, connected to terminal.
2. Remove handset from cradle; dial telephone number supplied by your system administrator.	2. Remove handset from cradle; dial telephone number supplied by your system administrator.
3. When you hear high-pitched tone, go on to Step 4. If line is busy or there is no answer, hang up and redial.	3. When you hear high-pitched tone, go on to Step 4. If line is busy or there is no answer, hang up and redial.
4. Press DATA button. DATA light should go on and keyboard unlock.	4. Place handset face down in coupler box; make sure cord is in slot. Close and latch coupler lid.
5. Place handset in cradle.	5. Turn on coupler switch within 20 seconds. Keyboard should unlock.
6. If DATA light goes out during session, retry from Step 1.	

Note: These procedures depend on the particular telephone data set or acoustic coupler you're using. See your system administrator.

Figure 8. Switched Line Procedures for Terminal Connection

For IBM start-stop terminals upon initial connection to the computer, it may be necessary to enter the following characters:

```
/"nn
```

where *nn* is a 2-digit number optionally predefined by your system administrator. Ask your system administrator whether or not this entry is required for your 2741 terminal.

Once you have made the connection with the computer, you are ready to contact VSPC by entering the VSPC logon command.

VPSC Command (Logon)

The VSPC command connects the terminal to VSPC.

VSPC	**ID=*usernum***	Required first operand.
	x	Required operand for S/S terminals, ignored for others.

ID=*usernum*
identifies you to VSPC. This number is assigned by your VSPC administrator. A usernum contains from one to seven digits; a value of zero (0) is not a valid user number.

x
identifies your terminal to VSPC. Type upshift 2. It is required for 3767 in S/S mode, 2741, and 1050; if you forget or make an error, VSPC will prompt for an upshift 2. For other VSPC terminals, it is ignored.

The VSPC command must be entered either entirely in uppercase letters or entirely in lowercase letters, as follows:

```
VSPC ID=12345
vspc id-12345
```

or as determined by your VSPC administrator.

Note: Depending on the default your organization has chosen, you may or may not be able to abbreviate the VSPC command. Check with your system administrator.

If a password is required, VSPC then prints the message:

```
ENTER PASSWORD
■■■■■■■
```

For security purposes, VSPC prints a series of overstruck characters (blot) over which you type your password. If the password you provide is not accepted, VSPC repeats the prompt and you can enter another password. If you do not provide the proper password within four attempts or two minutes, your connection is terminated.

When you have successfully logged on, some messages print at your terminal:

```
attribute date time usernum
[CONTINUE attribute[PASSWORD]]
[logon-message-text]
READY
```

Note: The portions of messages shown in brackets are not always printed.

- *attribute* designates your workspace attribute as specified in your profile (list of VSPC user characteristics); probably BASIC or LBASIC.
- *date time* displays the date and time you logged on.
- *usernum* states your user number.
- CONTINUE (when present) indicates that the contents of your workspace at the end of your last terminal session have been saved in the CONTINUE file and are available again with a LOAD CONTINUE command.

- *attribute* (in the second line) is the attribute of your CONTINUE file.
- PASSWORD is printed only if CONTINUE is protected with a file password. When you load the CONTINUE workspace, you must supply the correct password.
- *logon-message-test* (when present) is a general information message which may change periodically.
- READY indicates VSPC is ready to accept input.

If your logon is not successful, VSPC will print the reason. You then must repeat the procedure, correcting the indicated problem.

Example of 2741 or 3767 Start-Stop Logon

(With the 2741 terminal you may need to type some special keys before logging on to identify your terminal connection to the computer. See your VSPC administrator for instructions.)

You contact VSPC:

```
vspc id=1234567 @
```

VSPC prompts for your password and you type yours over the blot characters. (The @ sign represents an uppercase 2.)

```
ENTER PASSWORD
■■■■■■■
```

VSPC prints messages:

```
BASIC  02/04/78  09:16:20  1234567
CONTINUE BASIC
WELCOME TO VSPC
READY
```

Conversing with VSPC

When engaged in a conversation with VSPC, you tell VSPC what to do from your terminal, and VSPC responds. Responses from VSPC vary depending on the type of terminal and whether you are typing lines of data or are running a program.

This discussion applies to the 3767 in start/stop mode and 2741, with some reference to other terminals. For more detail, see *VS Personal Computing (VSPC) Terminals.*

1. VSPC notifies you in one or more of the following ways that it is ready to accept input:
   - Unlocks the keyboard
   - Displays a question mark (?)
   - Lights the ON LINE and PROCEED indicators
   - Prints a message (for example, READY)
   - Prints a line number (for example, 00010)

2. You type a line of input as follows:
   a. Type the line of information.
   b. Press carrier return.
   c. Wait for VSPC to respond in one or more of the above ways before your type another line.

Note: Some users of 2741, 1050, and CPT-TWX terminals should not enter '99999' as the first five characters of a terminal input request; some users of 3270 terminals should not respond by pressing the test request key. Either of these actions may cause the terminal to be disconnected from VSPC. See your VSPC administrator to determine whether these restrictions apply to your installation.

Correcting Errors

To correct typing errors in your current line:

1. Backspace to the error and signal attention.
2. VSPC responds with a series of actions:
   a. Prints an underscore (_) beneath the erroneous character.
   b. Backspaces one space.
   c. Advances one line and opens the keyboard.
3. Type the correction and the remainder of the line.
4. Press carrier return and wait for the VSPC response.

For example:

```
100 A=12E
        3
```

Deleting Lines

To delete an entire line you are typing:

1. Press attention without an immediately preceding backspace.
2. VSPC prints an asterisk (*), effects a carrier return, and opens the keyboard.

For example:

```
100 A=12A*
```

Use of Attention Key

In addition to character correction and line deletion, you can use the attention (ATTN) key for four other purposes:

- To signal the system—When your terminal is idle, pressing ATTN sends a signal to the system.
- To cancel output—When your terminal is printing, pressing ATTN ends the printing.
- To escape from your current input request—When it is the only response to an input request, pressing ATTN lets you cancel that request.

- To unlock your terminal for input when a MESSAGE WAIT command is in effect.

VS BASIC provides two modes for entering an attention during program execution: with an ON ATTN in effect or with ON ATTN not in effect.

If ON ATTN is in effect, you can hit the carriage return to begin execution at the point specified in the ON ATTN statement, or you can enter RES to resume execution at the point of the interruption. Any other action you take terminates execution of the program.

Invalid Output Characters

If you find an N backspace Z (X) character in your output, it is an indication that VSPC has found an invalid character in your data. On other terminals an invalid character may be shown by other symbols. See Appendix D.

PROGRAM DEVELOPMENT

A VS BASIC program is an organized plan created to solve a problem or produce a result. You send this plan to a computer, where each step is carried out in a specified order and some response is sent back to you. In between the creation and the successful running of this program you may change the content and move, add, delete, and renumber lines. To produce a program in VS BASIC under VSPC, you need to become familiar with some VSPC commands that make these actions possible.

The program as you create it in your workspace—the group of VS BASIC statements—is called a "source program." When you want to execute this program, you issue a RUN command. The RUN command invokes the VS BASIC compiler which translates your source program into a language the computer can understand. This translation process is called compilation. After your program is translated into machine language, the set of machine language instructions is called an object program. As an object program it is executable—when you issue a RUN command for an object program, the compilation is not repeated.

Several steps are necessary to create and execute a VS BASIC program.

1. First, you create your program line-by-line through the line entry procedure (optionally using the INPUT command for automatic line numbering). Then use the SAVE command to place it in a library as source code. With the SAVE command you can give the program a name by which it can be recovered. If you wish to have the new program replace a program that already exists in your library, you will have to use the NAME command to give it the same name.
2. You can modify your program with the various VSPC edit commands.
3. You execute your program using the RUN command; compile it and place it in a library as a machine language object program with the STORE command.

All these commands, plus others useful in VS BASIC program development are described in the following sections.

Creating a Program

When you have made connection with the computer and with VSPC by the logon procedure, and your workspace attribute is BASIC or LBASIC, you are ready to create a VS BASIC program.

Figure 9 shows how you can create a program named RANDOM, using the INPUT command to generate line numbers. (Assume you're using an IBM 3767 terminal in S/S mode or a 2741 terminal.)

The program, RANDOM, is a guessing game that generates a random number between 1 and 100 and asks the person at the terminal to guess the number. When all statements have been typed, you, as user number 1234567, press carrier return in response to line number 200. The system then types the word READY and gives you a new line on which to type your next entry. To protect your data from possible loss, issue the SAVE command, and the READY message appears again.

```
vspc id=1234567 @
ENTER PASSWORD
HHHHHHH
BASIC  01/04/76  09:16:20  1234567
WELCOME TO VSPC
READY

name random
READY
input
00010 print 'i''m thinking of a number between 1 and 100'
00020 print 'you have eight chances to guess my number'
00030 let n=int(rnd(0)*100+1)
00040 if t=8 goto 170
00050 let t=t+1
00060 print using 70,t
00070 :please make guess number
00080 input g rem accept the guess
00090 if g .eq. n goto 150
00100 if g .gt. n goto 130
00110 print 'no, my number is higher'
00120 goto 40 rem try again
00130 print 'no, my number is lower'
00140 goto 40 rem try again
00150 print 'congratulations—you guessed it'
00160 goto 190 rem terminate
00170 print using 180,n
00180 :sorry, you've had 8 guesses, my number was
00190 end
00200 (press carrier return)
READY
save
READY
```

Figure 9. Creating a VS BASIC Program

(The RANDOM program shown in Figure 9 will be used in later examples in this book.)

NAME Command

The NAME command assigns a name to your workspace. This name becomes the "filename" when a SAVE or STORE command is issued.

NAME	*filename*	Required.

filename
specifies the identification of your workspace. It must be a valid VSPC filename. (See "Filenames" section.)

If you specify an already existing filename with NAME, you are warned that your workspace will replace the contents of the library file with the new file, but the command is executed. (If you don't want to replace the old file, you simply issue another NAME command with a different *filename.*)

Examples:

```
name 1234567 random/jbl
```

assigns the filename 1234567 RANDOM to the contents of your workspace with a password, JBL.

```
nam random
```

assigns the filename 1234567 RANDOM to the contents of your workspace without a password.

```
name 456789 style
```

assigns the name STYLE to your workspace and will place it in project library 456789 when you issue a SAVE command.

Line Entry

Line entry adds data to your workspace and is represented in the form:

```
n entry
```

- n represents a line number. It can be from one to five digits long. Leading zeros need not be supplied.
- entry is what you type after the line number—the material you want processed. If not preceded by a line number, VSPC interprets a line entry as a command.

In VSPC you can enter data lines by typing the line number, an optional space, the data, and a carrier return (you can enter the lines in any numerical order). VSPC then unlocks the keyboard so that you can enter another line. (For this example, we'll assume your workspace is empty.) For example, you type:

```
40 input x1,y1
50 y=min (x1,y1)
60 if x<=0 | y<=0 then 90 else 100
70 print 'your values not valid; try again.'
```

Note that line numbers are in increments of at least 10. This gives you the ability to enter lines between already existing lines. For example, you decide to add one more statement to your program:

```
45 x=max (x1,y1)
```

However, VSPC treats the lines as if you'd specified them in the following order:

```
40 input x1,y1
45 x=max (x1,y1)
50 y=min (x1,y1)
60 if x<=0 | y<=0 then 90 else 100
70 print 'your values not valid; try again.'
```

VSPC inserts line 45 in numerical order between lines 40 and 50.

You can replace a line you've already entered simply by entering a new line with the same line number. For example, if you type:

```
70 print 'invalid values; try again.'
```

then this new line 70 completely replaces the original line 70, even though the new line 70 is shorter than the original line 70.

You can delete an existing line by typing the line number followed immediately by a carrier return.

When n is less than five digits long and the first character of your entry is numeric, a space must be inserted between n and your entry. Otherwise you can omit the space.

Examples:

To add data to the end or insert lines into your program in Figure 0, type the line number and the line entry. For example,

```
25 print 'follow my directions'
```

will insert a line between lines 20 and 30.

```
190 print 'better luck next time'
```

will replace END; then, to properly end your program, type

```
200 end
```

INPUT Command

The INPUT command automatically numbers your input data lines for you.

INPUT	*begin-line* *begin-line increment* *	Optional first operand; choose one or omit. If omitted, for empty workspace line 10 is assumed; for nonempty workspace next higher multiple of 10 than last existing line is assumed; increment of 10 is assumed.
	OVERLAY	Optional; specify or omit. If omitted, existing lines are not replaced.
	PROMPT **NOPROMPT**	Optional; choose one or omit. If omitted, PROMPT is assumed.

begin-line
specifies the first line number from 0 through 99999 to be generated.

If that line number already exists and the OVERLAY option was not specified, the line number entered will be the first number after that specified.

increment
specifies the increment for succeeding line numbers from 1 through 99999.

*
denotes your current line. It may be specified instead of *begin-line*.

OVERLAY
specifies that entered lines are to overlay existing lines if they have the same line numbers.

PROMPT
specifies that automatically generated line numbers will be displayed at the terminal.

NOPROMPT
specifies that automatically generated line numbers will not be displayed at the terminal.

When you don't specify any operands and your workspace is empty, the first line number generated is 10, and each succeeding line number will be incremented by 10.

When your workspace contains data and you don't specify any operands, the first generated line number is the next higher multiple of 10 after your last line and each succeeding line number is incremented by 10.

When you specify *begin-line,* this line number is the first one generated. If *increment* is omitted, succeeding line numbers are incremented by 10. If *increment* is specified, that increment is used for succeeding line numbers.

When your workspace contains a BASIC source program, the INPUT command is terminated when the next line number would cause overlap. You can then reissue the command with a different *begin-line* to avoid loss of lines.

Note: You cannot backspace and alter a generated line number.

Example:

```
input * prompt
```

in VIEW mode formats the screen with line numbers, starting with your current line, in the first five character positions (as shown in Figure 7).

When you specify NOPROMPT, a message prints

```
LINE n
```

where n is the first generated line number. No other line numbers print.

When you want to end automatic line numbering, you press carrier return before entering any data on a line. The response differs, depending on which prompting option you have specified:

- When you have specified the PROMPT option, VSPC displays the READY message, and opens the keyboard for command or line entry.
- When you have specified NOPROMPT, VSPC prints:

  ```
  INPUT ENDED AT LINE n
  ```

  where n is the last number generated. The keyboard unlocks for command or line entry.

Examples:

```
input 310 5 noprompt
```

automatically numbers lines beginning with line 310, increments succeeding lines by 5, and does not display line numbers at the terminal.

```
inp
```

automatically numbers lines if your workspace is empty beginning with the default line 10, increments by the default of 10, and displays the numbers (PROMPT is the default). However, if your workspace contains a VS BASIC program whose highest line number is 185, then line numbering starts at 190 and increments succeeding lines by 10.

SAVE Command

The SAVE command places a copy of your workspace in a library. (The SAVE command lets you save VS BASIC source programs.)

SAVE	*filename*	Optional; specify or omit. If omitted, workspace is saved under current name.

filename
specifies the identifying *filename* to be used. Your workspace is also renamed with this name. Your *filename* must conform to the rules for VSPC filenames. (See "Filenames" section.)

Assuming you have previously assigned a name to your workspace, the SAVE command sends a copy of the contents of your workspace to the library you specify, while retaining it in your workspace. Your workspace name is changed to *filename*.

If you issue a SAVE without the *filename* operand and your workspace is unnamed, you will get a message:

```
WORKSPACE NOT NAMED
```

and the SAVE command is not executed. You can then use the filename operand to name your workspace.

```
save random
```

Alternatively, you can first name your workspace and then use the SAVE command without an operand:

```
name random
save
```

The filename, including the library number and name, is used to identify your file. If you have named your program RANDOM and issue the SAVE command with no operands, your program is sent to your own library with the filename 1234567 RANDOM.

However, if PROG is the name of your workspace, you can change it with the SAVE command.

You can rename it and send it to your library:

```
save random
```

You can send it to your project library using either of the following forms:

```
save 456789 random
```

or

```
save 0 random
```

You can attach a password to it and send it to your library:

```
save random/cat
```

When you add a password to your filename, you are required to supply it before you can retrieve the file.

Renaming your files is useful in program development. For example, if you've named a program, developed it, and saved it, but you want to try another method of solution, you can make changes, rearrange it, and then, with the

SAVE command, send a copy of it to your library with a different name. This way you will have both programs to work with.

If you specify an already existing filename with SAVE, your command is rejected, unless your workspace already has the same filename. The attribute of the workspace and the file must be the same.

You can only save a CONTINUE file in your own library. An existing CONTINUE file is overlaid even if the attributes are different.

Examples:

```
save 456789rain/wet
```

saves your workspace contents in your project library, with the name RAIN, and the password WET.

```
sav
```

sends the contents of your workspace to your own library with the name you've previously assigned.

CLEAR Command

If you create a program or part of a program and don't want to save it, you can empty your workspace with the CLEAR command.

CLEAR		No operands allowed.

The CLEAR command erases all lines, including any filename, PROTECT, or SHARE specification. Your workspace retains whatever attribute it had (BASIC, LBASIC).

Examples:

```
clear
```

erases your workspace.

```
cle
```

erases your workspace.

Modifying a Program

If you issued a SAVE command on your program at a previous session, you will need to get it back into your workspace with the LOAD command in order to modify it. Before using any other commands, it's a good idea to display your program with the LIST command so that you know exactly what it contains.

Use FIND to search your workspace for sets of characters and display them on your terminal screen.

You can modify your source program using the CHANGE, DELETE, MOVE, COPY, RENUMBER, MERGE, EXTRACT, JOIN, and SPLIT commands.

Use the TEXT command to tell VSPC whether or not to display the text of the line affected by your CHANGE, FIND, and LOCATE commands.

LOAD Command

The LOAD command places a specified file in your workspace.

LOAD	*filename*	Required.

filename
specifies which file to place in the workspace; *filename* must conform to the rules for VSPC filenames. (See "Filenames" section.)

When you issue a LOAD command, the attribute of your workspace is changed (if necessary) to match the newly loaded file. The current contents of your workspace are completely replaced. All workspace characteristics, such as name, attribute, access control options, and numbering conventions, are changed to match the new contents.

You can load a file containing a command list, a data file, or a compiler source program. You can't load an object program or an undefined file with the LOAD command.

If you load a program owned by someone else that is protected with a PROTECT NOREAD, then you can run it but not list or save it. See "PROTECT Command."

Examples:

You, as user number 1234567, can load the program called RANDOM into your workspace from your private library.

```
loa random/cat
```

The complete filename of RANDOM is 1234567 RANDOM with the password CAT. Anyone else attempting to access it must use the entire filename and the password.

```
load 1234567 random/cat
```

Suppose you execute the following sequence of commands:

```
load 456789 style/pass
save design
```

This brings a file named STYLE with password PASS from your project library (456789) into your workspace and then puts it in your own library with the filename 1234567 DESIGN. The same program still exists in libnum 456789 as STYLE with the password PASS. If DESIGN already exists in your library, you must use the NAME command to assign the name DESIGN to your workspace and then issue SAVE to replace the already existing file named DESIGN.

LIST Command

The LIST command displays all or part of the contents of your workspace.

LIST	*line* *line*:*end-line* *	Optional first operand; choose one or omit. If omitted, all lines are displayed.
	LINE **NOLINE**	Optional; choose one or omit. If omitted, LINE is assumed.

line
specifies the line you want to display.

line:*end-line*
specifies the range of lines to be displayed.

*
specifies your current line. It can denote either *line* or *end-line*.

When no line number is specified, all lines in your workspace are displayed.

LINE
specifies that the line numbers are to be displayed with the listing. When neither LINE nor NOLINE is specified, LINE is assumed.

NOLINE
specifies that the line numbers are not to be displayed with the listing.

If you do not own the file currently in your workspace, and the file has PROTECT NOREAD in effect, your LIST command is not executed and you receive a message.

Examples:

To display RANDOM, which is in your workspace, you type

```
list
```

See Figure 10 for the system response.

```
 10 PRINT 'I'M THINKING OF A NUMBER BETWEEN 1 AND 100'
 20 PRINT 'YOU HAVE EIGHT CHANCES TO GUESS MY NUMBER'
 30 LET N=INT(RND(0)*100+1)
 40 IF T=8 GOTO 170
 50 LET T=T+1
 60 PRINT USING 70,T
 70 :PLEASE MAKE GUESS NUMBER
 80 INPUT G REM ACCEPT THE GUESS
 90 IF G .EQ. N GOTO 150
100 IF G .GT. N GOTO 130
110 PRINT 'NO, MY NUMBER IS HIGHER'
120 GOTO 40 REM TRY AGAIN
130 PRINT 'NO, MY NUMBER IS LOWER'
140 GOTO 40 REM TRY AGAIN
150 PRINT 'CONGRATULATIONS—YOU GUESSED IT'
160 GOTO 190 REM TERMINATE
170 PRINT USING 180,N
180 :SORRY, YOU'VE HAD 8 GUESSES, MY NUMBER WAS
190 END
```

Figure 10. Listing of Program

You can list one line with its line number (LINE is the default):

```
list 50
```

The system responds:

```
50 LET T=T+1
```

You can list one line *without* the line number:

```
list 50 noline
```

The system responds:

```
LET T=T+1
```

You can list a group of lines with line numbers:

```
lis 80:100
```

The system responds:

```
 80 INPUT G
 90 IF G=N GOTO 150
100 IF G>N GOTO 130
```

You can list a group of lines without line numbers:

```
lis 40:50 nol
```

The system responds:

```
IF T=8 GOTO 170
LET T=T+1
```

FIND Command

The FIND command searches your workspace for a particular set of characters.

FIND	*line* *line*:*end-line* *	Optional first operand; choose one or omit. If omitted, all lines are searched.
	'string '	Required.
	TEXT **NOTEXT**	Optional; choose one or omit. If omitted, NOTEXT is assumed; if specified, must follow *'string '* operand.

line
specifies a single line to be searched; it must exist in the workspace.

line:*end-line*
specifies a range of lines to be searched; *end-line* must be equal to or greater than *line*. Neither line need actually exist in the workspace.

*
specifies your current line. It can denote either *line* or *end-line*.

'string '
specifies the group of characters you are looking for. It can consist of 0 through 255 characters. A string must be enclosed in single quotation marks. If a single quotation mark is contained within the string, it must be specified as two adjacent single quotation marks.

TEXT
specifies that lines containing the *'string '* and their line numbers are to be displayed at the terminal.

NOTEXT
specifies that only the numbers of the lines in which the *'string '* is found are to be displayed; not the text of the lines. If TEXT and NOTEXT are both omitted, NOTEXT is in effect unless you used the TEXT ON command.

If line numbers specified in the range do not exist, all existing numbers within the range are searched.

The FIND command places the found line at the top of the screen (but does not change the current line pointer).

Examples:

You want to search lines 35 through 95 of your RANDOM program for the word 'PRINT', and so you can specify:

```
find 35:95 'print' text
```

The system responds

```
60 PRINT USING 70,T
```

because line 60 is the only line between 35 and 95 with the word 'PRINT'.

Now you specify

```
fin 'goto'
```

and the entire RANDOM program is searched for 'GOTO'. VSPC prints only the numbers of the lines containing 'GOTO':

```
40 90 100 120 140 160
```

CHANGE Command

The CHANGE command modifies data in your workspace.

CHANGE	*line* *line*:*end-line* * **ALL**	Required first operand; choose one.
	'string-1 ' *'string-1 ' 'string-2 '*	Required; choose one.
	TEXT **NOTEXT**	Optional; choose one or omit. If omitted, NOTEXT is assumed; if specified, must follow *'string '* operand.

line
specifies the one line in which to make a change. The line number specified must exist in your workspace.

line:*end-line*
specifies a range of lines to be searched. The *end-line* must be equal to or greater than *line*. The line numbers specified need not actually exist; all existing lines within the specified range are searched.

*
specifies your current line. It can denote either *line* or *end-line*.

ALL
specifies that all lines in the workspace be searched for *'string '*.

'string-1 '
specifies the group of characters you want replaced. It can consist of 0 through 255 characters and is enclosed in single quotation marks.

'string-2 '
specifies the group of replacement characters and can consist of 0 through 255 characters enclosed in single quotation marks. String-2 need not contain the same number of characters as string-1.

In both strings all characters are valid. However, a single quotation mark within either string must be specified as two adjacent quotation marks (' '), which will appear in your workspace as one character.

TEXT
specifies that changed lines are to be displayed at the terminal.

NOTEXT
specifies that changed lines are not to be displayed. When TEXT and NOTEXT are omitted, the last specification of the TEXT command is in effect.

On a 3270 Display System terminal you can issue the CHANGE command with only the line number. VSPC responds with a display of the line and you can change it (type over it, use DEL or INS MODE) and press ENTER.

When you issue the CHANGE command, all indicated strings within the range specified are changed. If no range is specified, all lines are searched. If you omit *string-2*, *string-1* will be removed.

The current line pointer is unchanged unless you are using a display terminal. Enter

```
change  line
```

and then reenter the displayed line. In this case the reentered line becomes your current line.

If you enter two adjacent single quotation marks as *string-1*, it is assumed to be an empty string and *string-2* is added at the end of each line in the specified range. If there is no *string-2* in this case, all lines or line numbers (depending on whether TEXT or NOTEXT is in effect) within the specified range are displayed.

Examples:

You can specify a change to be made on every line of your program.

```
change all 'number' 'figure'
```

The system responds:

```
10  20  70  110  130  180
6 LINES CHANGED
```

If you want to change 'guesses' to 'chances' in line 180, and to have the lines printed, you type the command as follows:

```
change 180 'guesses' 'chances' text
```

The system responds:

```
180 :SORRY, YOU'VE HAD 8 CHANCES, MY FIGURE WAS
1 LINES CHANGED
```

DELETE Command

The DELETE command eliminates lines from your workspace.

DELETE	*line* *line*:*end-line* *	Required; choose one.

line
specifies one line to be deleted. It must exist in the workspace.

line:*end-line*
specifies a range of lines to be deleted. Neither line number need actually exist. The *end-line* must be equal to or greater than *line*.

*
specifies your current line. It can denote either *line* or *end-line*.

To be sure your line numbers are accurate, check them with the LIST command, before using DELETE.

Examples:

You can delete a line or group of lines.

```
delete 100
```

deletes line number 100 and the system responds with the number of lines deleted:

```
1 LINES DELETED
```

You can delete lines 20 through 70.

```
delete 20:70
```

The system responds:

```
6 LINES DELETED
```

To delete lines 20 *and* 70 and not those in between, you must use the DELETE command twice:

```
delete 20
```

The system responds:

```
1 LINES DELETED
```

Then enter

```
delete 70
```

The system responds:

```
1 LINES DELETED
```

MOVE Command

The MOVE command rearranges lines within your program.

MOVE	*line* *line*:*end-line* *	Required first operand; choose one.
	destination-line *	Required second operand.
	increment	Optional; specify or omit. If omitted, 10 is assumed.

line
specifies one line to be moved. The line number must exist in your workspace.

line:*end-line*
specifies a range of lines to be moved. If any line number specified in a range does not exist, then existing lines within the range specified are moved. The *end-line* must be equal to or greater than the *line*. Line numbers can be between 0 and 99999.

*
specifies your current line. It can denote either *line*, *end-line*, or *destination-line*.

destination-line
specifies the line number after which the earlier specified lines are to be moved; it need not actually exist in the workspace. It may be higher or lower but not within the range.

increment
specifies the increment for renumbering the lines involved in the move; must be between 1 and 99999.

If the last generated line number overlaps the line numbers after the inserted lines, the original lines following the insert are renumbered as far as is necessary to maintain the ascending sequence, and you receive a message:

```
RENUMBERED FROM LINE n
```

where n is the number of the first original line to be renumbered.

Line references in your VS BASIC source program are adjusted to match the newly assigned numbers.

Examples:

The command:

```
move 70 50
```

moves line 70 to after line 50, increments by 10, and responds:

```
RENUMBERED FROM LINE 60
1 LINES MOVED
```

The change looks like this:

Before MOVE

```
10 LET P1=39
20 LET N1=12
30 LET P2=47
40 LET N2=14.5
50 LET K1=P1/N1
60 LET K2=P2/N2
70 PRINT K1
80 PRINT K2
90 END
```

After MOVE

```
10 LET P1=39
20 LET N1=12
30 LET P2=47
40 LET N2=14.5
50 LET K1=P1/N1
60 PRINT K1
70 LET K2=P2/N2
80 PRINT K2
90 END
```

If you issue the command:

```
move 30:40 60 5
```

VSPC places lines 30 and 40 after line 60, and renumbers the succeeding lines in increments of 5, as follows: line 30 becomes line 65, line 40 becomes line 70, line 70 becomes line 75. The original line 80 is in the correct ascending sequence, and so its line number is not changed.

The system responds:

```
RENUMBERED FROM LINE 70
2 LINES MOVED
```

The lines are rearranged as follows:

Before MOVE

```
10 LET P1=39
20 LET N1=12
30 LET P2=47
40 LET N2=14.5
50 LET K1=P1/N1
60 PRINT K1
70 LET K2=P2/N2
80 PRINT K2
90 END
```

After MOVE

```
10 LET P1=39
20 LET N1=12
50 LET K1=P1/N1
60 PRINT K1
65 LET P2=47
70 LET N2=14.5
75 LET K2=P2/N2
80 PRINT K2
90 END
```

Note that when your MOVE command causes renumbering, affected lines referenced within your program are changed also:

Before MOVE

```
10 PRINT 'REPLY YES TO CONTINUE, NO TO END JOB'
20 IF A$='NO' GOTO 80
30 INPUT A,B,C
40 D=(A+B+C)/3
50 INPUT A$
60 PRINT 'AVERAGE IS';D
70 GOTO 50
80 END

move 50 10
```

After MOVE

```
10 PRINT 'REPLY YES TO CONTINUE, NO TO END JOB'
20 INPUT A$
30 IF A$='NO' GOTO 80
40 INPUT A,B,C
50 D=(A+B+C)/3
60 PRINT 'AVERAGE IS';D
70 GOTO 20
80 END
```

Now you issue the following command:

```
move 40:90 60 10
```

This requests VSPC to move lines 40 through 90 to follow line 60. This is impossible because line 60 is within the group to be moved. The system responds with an error message:

```
DESTINATION WITHIN MOVE RANGE
```

COPY Command

The COPY command duplicates an existing line or range of lines in another location in the workspace.

COPY	*line* *line*:*end-line* *	Required first operand; choose one.
	destination-line *	Required second operand.
	increment	Optional third operand; specify or omit. If omitted, 10 is assumed.

line
specifies one line to be copied. The line number must exist in your workspace.

line:*end-line*
specifies a range of lines to be copied. If any line number specified in a range does not exist, then existing lines within the range specified are copied. The *end-line* must be greater than or equal to *line*. Line numbers can be between 0 and 99999.

destination-line
specifies the line number after which the lines are to be copied. It need not exist in your workspace; it may be higher or lower, but not within the range of lines being copied. If it does not exist, it becomes the number of the first copied line.

*
specifies your current line. It can denote *line*, *end-line*, or *destination-line*.

increment
specifies the increment by which the copied lines are to be numbered. It must be an integer between 1 and 99999. When *increment* is omitted, an increment of 10 is assumed.

If the newly inserted line numbers overlap existing numbers, the lines following the insert are renumbered with the same increment as far as is necessary to maintain the line number sequence.

If the destination line is within the range of lines to be copied, the command is rejected.

The rules that apply to MOVE also apply to COPY.

Examples:

```
copy 290 400
```

copies line 290 with the number 00400 at line 00400. If necessary, lines are renumbered and a message returned:

```
LINES RENUMBERED FROM 410
```

```
copy 290:310 400 2
```

copies lines 290 through 310, places them at line 400, and increments by 2: 00402, 00404, 00406... . No renumbering is done.

RENUMBER Command

The RENUMBER command reassigns line numbers in your workspace.

RENUMBER	* *old-line* *old-line new-line* *old-line new-line increment*	Optional; choose one or omit. If omitted, all lines renumbered with increment of 10.

*

specifies your current line. It can denote *old-line* or *new-line*.

old-line

identifies a line number in your workspace at which to start the renumbering. It must have a value from 0 through 99999.

new-line

specifies the new value to assign to *old-line*. The *new-line* must have a value from 0 through 99999.

increment

specifies the increment to be used in assigning new line numbers; it must have a value from 1 through 99999.

If all operands are omitted, all lines of data in the workspace are renumbered. The first line is assigned line number 10; succeeding line numbers are incremented by 10.

If only *old-line* is specified, the data beginning at *old-line* and continuing to the end of the data are renumbered. The new line numbers are multiples of 10, and are incremented by 10.

If *old-line* and *new-line* are specified, then *old-line* (the old line number) is replaced by *new-line* (the new line number), and succeeding line numbers are incremented by 10. The new line number must be higher in value than the last line number preceding the old line number; that is, there must be no overlapping of old and new line numbers.

If the value of *old-line* falls between the values of existing line numbers in the workspace, the next higher existing line number is the first line renumbered.

If *increment* is specified, then the new line numbers are incremented by the value specified.

Examples:

```
renumber
```

renumbers your entire workspace starting with line 10 in increments of 10.

Before RENUMBER

```
10 LET P1=39
20 LET N1=12
50 LET K1=P1/N1
60 PRINT K1
65 LET P2=47
70 LET N2=14.5
75 LET K2=P2/N2
80 PRINT K2
90 END
```

After RENUMBER

```
10 LET P1=39
20 LET N1=12
30 LET K1=P1/N1
40 PRINT K1
50 LET P2=47
60 LET N2=14.5
70 LET K2=P2/N2
80 PRINT K2
90 END
```

To renumber line 50 as line 100, and to increment succeeding lines by 5, you specify:

```
ren 50 100 5
```

VSPC responds

```
5 LINES RENUMBERED
```

In your workspace, the lines are rearranged as follows:

Before RENUMBER

```
10 LET P1=39
20 LET N1=12
30 LET K1=P1/N1
40 PRINT K1
50 LET P2=47
60 LET N2=14.5
70 LET K2=P2/N2
80 PRINT K2
90 END
```

After RENUMBER

```
 10 LET P1=39
 20 LET N1=12
 30 LET K1=P1/N1
 40 PRINT K1
100 LET P2=47
105 LET N2=14.5
110 LET K2=P2/N2
115 PRINT K2
120 END
```

Note that when your program is renumbered, affected lines referenced within your program are changed also: for example, if you type your program with increments of 5, as follows:

Before RENUMBER

```
 5 PRINT 'REPLY YES TO CONTINUE, NO TO END JOB'
10 INPUT A$
15 IF A$='NO' GOTO 40
20 INPUT A,B,C
25 D=(A+B+C)/3
30 PRINT 'AVERAGE IS';D
35 GOTO 10
40 END
```

and then decide to renumber the program with increments of 10 (renumber or renumber 10 10), the system will change the line numbers referenced in lines 15 and 35 (now 30 and 70) to match the renumbered lines of your program.

After RENUMBER

```
10 PRINT 'REPLY YES TO CONTINUE, NO TO END JOB'
20 INPUT A$
30 IF A$='NO' GOTO 80
40 INPUT A,B,C
50 D=(A+B+C)/3
60 PRINT 'AVERAGE IS';D
70 GOTO 20
80 END
```

MERGE Command

The MERGE command retrieves a library file and combines it with the contents of your workspace.

MERGE	*filename*	Required first operand.
	* *destination-line* *destination-line increment* **OVERLAY**	Optional; choose one or omit. If omitted, file placed at end of workspace contents.

filename
identifies the library file to be retrieved. It must contain editable data and conform to the rules for VSPC filenames (see "Filenames" section).

*
specifies your current line. It can denote your *destination-line*.

destination-line
identifies the line in the workspace after which the file is to be inserted; it must have a value from 0 through 99999.

increment
specifies the increment to be used for the inserted file line numbers; it must have a value from 1 through 99999. When *incr* is omitted, an increment of 10 is assumed.

OVERLAY
specifies that the file and the workspace are to be combined based on their current line numbers. If a line number in the file is the same as one in the workspace, the file line replaces the workspace line.

If you specify MERGE filename with no other operands, the file is placed at the end of your workspace and renumbered in increments of 10 from the last current line. The line number given the first added line is the next multiple of 10 higher than any existing line number.

If you want to place the file somewhere other than at the end of the workspace, you can specify a line number after which the file is to be inserted. In this case, the inserted file will be renumbered in increments of 10 or to your specification. If the destination line specified does not exist, the next lower existing line number in your workspace is assumed.

If the last generated line number overlaps the next existing line number after the insertion, the original lines following the insert are renumbered as far as is necessary to maintain the ascending sequence, and you receive a message:

```
RENUMBERED FROM LINE n
```

where n is the number of the first original line to be renumbered.

With OVERLAY you can interlace the lines of the two programs.

Warning: If any line numbers are identical, the new file lines will replace the workspace lines.

The attribute of your workspace is not changed. If the attribute of the inserted file is different from the workspace attribute, your MERGE command is completed and a message is issued:

```
WARNING - DIFFERENT FILE TYPE MERGED
```

If the inserted file has a PROTECT NOREAD characteristic and you do not own it, your workspace assumes that protection after the MERGE command. This means, for example, that you could not execute the LIST command on this file.

See *VS Personal Computing (VSPC) General User's Guide and Command Language* for details concerning differing file types and security protection commands as they affect merging.

Examples:

```
merge 456789 style/pass
```

retrieves a file named 456789 STYLE/PASS from your project library and places it after the current contents of your workspace.

```
mer car
```

retrieves your file named CAR and adds it to the current contents of your workspace.

```
merge car 40 5
```

gets CAR from your library and inserts it after line 40 of your current program, renumbering the lines in increments of 5.

The system responds to alert you that your line numbers are different:

```
LINES RENUMBERED FROM 50
```

If you have RANDOM in your workspace and want to lace CAR into it:

```
mer car overlay
```

combines RANDOM with CAR according to the ascending order of their line numbers, and replaces any lines in RANDOM with identically numbered lines in CAR.

EXTRACT Command

The EXTRACT command clears your workspace leaving the specified lines.

EXTRACT	*line* *line:end-line* *	Required; choose one.

line
specifies a single line to be retained in the workspace. The line number must be in the workspace.

line:end-line
specifies the range of lines to be retained in the workspace. The *end-line* must be greater than or equal to *line*.

The line numbers specified need not be in the workspace; if they are not, existing lines within the range specified are retained.

*
specifies your current line. It can denote either *line* or *end-line*.

When EXTRACT is executed, the name of the resulting workspace is erased but the attribute is unchanged. The line numbers of the remaining lines are unchanged.

The current line pointer is unchanged if the current line is within the part of the workspace being kept. If the current line preceded the part retained, it is set to the first line of the resulting workspace. If the current line was after the retained part, it is set to the last line of the resulting workspace.

Examples:

```
extract 190
```

erases your entire workspace except line 00190.

```
extract 20:80
```

erases every line except 00020 through 00080.

JOIN Command

The JOIN command unites two consecutive lines of the workspace.

JOIN	*line* *	Required; choose one.

line
specifies the first line number of the two consecutive lines to be joined.

*
specifies that your current line be joined to the following line.

The joined line has the line number of the first of the joined lines. The line number of the second line is erased; succeeding lines are not renumbered.

Your current line number is unchanged when using the JOIN command unless the second line joined is your current line. In this case the joined line becomes your current line.

SPLIT Command

The SPLIT command divides a single line into two consecutive lines.

SPLIT	*line* *	Required first operand; choose one.
	number *'string '*	Required second operand; choose one.

line
specifies the number of the line to be split.

*
specifies that your current line be split.

number
specifies the number of characters to be included in the first line. (The line number and following blank are not counted.) If *number* is specified as zero, the first resulting line contains no characters. If *number* is greater than the number of characters in the line, the command is rejected.

'string '
specifies a set of characters that is to start the second line.

If *'string '* occurs more than once in *line*, the first one is assumed.

The line number to be given to the second line is calculated from the line number of the line being split and the number of the line which originally followed it in the workspace.

- If the line being split is the last line of the workspace or the line following it has a line number 20 or more higher, then the second line is given a number 10 higher than the original line.
- If the original interval was less than 20, the second line will have a number midway between the two original lines.
- If the line that originally followed the line being split has a number only one higher than the line being split, then the second line is given a number one higher than the original line and further lines are renumbered with an increment of one as far as is necessary to preserve the sequence of lines. You will be informed if this renumbering is performed:

  `LINES RENUMBERED FROM` *n*

  where *n* is the first changed line number.

TEXT Command

The TEXT command tells VSPC whether or not to display the text of the lines affected by the CHANGE, FIND, and LOCATE commands.

TEXT	**ON** **OFF**	Required; choose one.

ON
specifies that the text of lines affected by CHANGE, FIND, and LOCATE are to be displayed unless the command itself specifies NOTEXT.

OFF
specifies that the text of lines affected by CHANGE, FIND, and LOCATE are not to be displayed unless the command itself specifies TEXT.

Specification of the TEXT and NOTEXT operands in CHANGE, FIND, and LOCATE commands overrides this TEXT command. When you log on, NOTEXT is assumed.

Compiling and Executing a Program

After you have created a program and made modifications, the next step is to execute (run) it to see if it produces the desired results. If it does, then you can send it to a library as object code with the STORE command and run it the next time directly from the library without recompiling.

RUN Command

The RUN command compiles and/or executes a program or command list.

RUN	*filename*	Optional; specify or omit.
	'string '	Optional; specify or omit.

filename
specifies a file to be processed; *filename* must conform to the rules for VSPC filenames (see "Filenames" section).

'string '
specifies a character string to be passed to the program. It may consist of 0 through 255 characters enclosed in single quotation marks. All characters are valid. However, if a single quotation mark (') is contained within the string, it must be specified as two adjacent single quotation marks ('').

When the RUN command is executed, one of the following actions occurs:

- If *filename* is omitted, the compiler source program in the workspace is compiled and executed, or the CLIST in the workspace is executed.
- If *filename* specifies a compiler source program, the program is loaded, compiled, and executed.
- If *filename* specifies a compiler object program, the current data in the workspace is erased and the object program is executed.
- If *filename* specifies a command list, the list is loaded and executed.

When *filename* is omitted, the RUN command operates on the contents of the workspace. In this case, the attribute of the workspace must be BASIC, LBASIC, or CLIST.

When the RUN command successfully operates upon a compiler program, the program is compiled and/or executed. A program header is printed, followed by any compiler messages, followed by terminal interactions with the executing program (input or output).

When the RUN command successfully executes a command list, VSPC executes (and prints or suppresses printing of) the commands in the list (see "Command List (CLIST) Processing" section).

When a CLIST contains a RUN command naming a second CLIST, return is made to the next command in the first CLIST following completion of the second CLIST.

If you stored your RANDOM source program with the STORE command (using the name RANDOBJ), it was compiled and stored in your library as machine language object code under the name RANDOBJ. If you saved RANDOM with SAVE, it was placed in your library as source code under the name RANDOM. In either case, you can execute it with the RUN command using the correct filename.

Example:

(The changes used as examples in the preceding command descriptions are not reflected in this example.)

```
run random
```

VS BASIC compiles the program, and the system then executes it.

```
run randobj
```

The system erases the current data in your workspace and executes the object program.

In either case, during execution, the messages (and the terminal responses) are as shown in Figure 11.

```
RANDOM          1/29/76          3:40:31
I AM THINKING OF A NUMBER BETWEEN 1 and 100
YOU HAVE EIGHT CHANCES TO GUESS MY NUMBER
PLEASE MAKE GUESS NUMBER 1
?
20
NO, MY NUMBER IS HIGHER
PLEASE MAKE GUESS NUMBER 2
?
30
NO, MY NUMBER IS HIGHER
PLEASE MAKE GUESS NUMBER 3
?
45
NO, MY NUMBER IS LOWER
PLEASE MAKE GUESS NUMBER 4
?
42
NO, MY NUMBER IS LOWER
PLEASE MAKE GUESS NUMBER 5
?
37
CONGRATULATIONS—YOU GUESSED IT
TIME 2.3 SECS
```

Figure 11. Running a VS BASIC Program

STORE Command

The STORE command compiles the program in your workspace and files it in a library as machine language object code.

STORE	*filename*	Required.

filename
specifies the name of the file in which the object program is to be placed; it must conform to the rules for VSPC filenames (see "Filenames" section). CONTINUE must not be specified as the name.

You can use the STORE command to create a new object program or to replace one that already exists.

When the STORE command is executed, the source program in the workspace is compiled. If the compilation is successful, the resulting object program is placed in the specified file; it is identified by *filename.* If the file already exists, it must be an object file. After execution is completed, the source program in the workspace is unchanged.

Only a file's owner may replace an existing object program.

If the STORE command creates a new object program, that object program can be placed in your private or project library, or any public library. You use the STORE command when you have your program running in your workspace and want to store it as object code so it will run in the future without having to be recompiled. The source code in your workspace remains unchanged when you issue the STORE command.

During STORE command execution, an attention signal causes compilation to be terminated with an error message.

If you press the attention key while STORE is executing (before system response), you will escape from the command; it will not execute.

If there are severe errors in your source program, VSPC displays the filename; a series of compiler error messages follow, telling you what is wrong. VSPC then sends you the following message:

```
COMPILED PROGRAM NOT STORED
```

followed by a message showing the time taken for compilation.

When compilation is successful, the system responds with an identification header, the size of the object program, and the CPU time needed for compilation.

Examples:

```
store 456789 ranprog/pass
```

stores the contents of your workspace in your project library with the name RANPROG and password PASS.

```
sto ranprog
```

stores your workspace in your private library under the name RANPROG.

```
RANPROG 1/29/76 11:40:31
COMPILED SIZE 33135 BYTES
TIME 2.1 SECS
```

Note: Two additional operands ('LINK' and 'NOLINK') can be used with STORE in VSPC FORTRAN. They are ignored in VS BASIC.

Interrupting VS BASIC

There is a type of interrupt called an asynchronous interrupt that can occur at any time and is not connected with your BASIC program. Four such interrupts are:

- The attention interrupt, which is a request to communicate with VS BASIC.
- Cancel output, which is a request to cancel all output to the terminal.
- Processing time limit, which may or may not be specified in your user profile. If it is, it is a limit on the amount of processing time you can use between terminal interactions. It is included as the CPUMAX field when you issue a QUERY PROFILE command.
- Session termination, which occurs when you are forced to sign off from VSPC; for example, when your telephone connection to the system is broken.

The first two interrupts described above are the results of your actions. Some terminals have two interrupt buttons—one for attention and one for cancel.

Others have only one, marked ATTN or BREAK. In this case, pressing the button causes both an attention and a cancel output interrupt to occur.

When session termination occurs, the VSPC supervisor attempts to save your editable workspace under the name CONTINUE.

Interrupts During Compilation of Your VS BASIC Program

Attention

An attention interrupt causes VS BASIC to terminate processing with the message:

```
ICD091 COMPILATION TERMINATED BY ATTENTION
```

Cancel Output

A cancel output interrupt causes VS BASIC to suppress all output to the terminal. This status is reset:

- at the start of execution
- when you press attention

If execution is inhibited, a final message is sent to the terminal:

```
ICD092 SEVERE SOURCE ERRORS DURING COMPILATION
```

Processing Time Limit

This is ignored during the compilation of your program.

Session Termination

Compilation is terminated immediately.

Interrupts During Execution of Your BASIC Program

Attention

VS BASIC provides two modes for entering an attention: with ON ATTN in effect or with ON ATTN not in effect. If ON ATTN is in effect, you can hit the carriage return to begin execution at the point specified in the ON ATTN statement, or you can enter RES to resume execution at the point of the interruption. Any other action you take terminates execution of the program.

If ON ATTN is not in effect, an attention causes VS BASIC to interrupt execution of your program, and sends you the message:

```
ICD411 LINE xxxxx ATTENTION INTERRUPT
```

You are then prompted for input from your terminal. This message is sent immediately before any output queued by VS BASIC is sent to the terminal. If you respond with a null line (carrier return only), execution continues; if you enter any data or press attention, execution is terminated and the message is sent to the terminal:

```
ICD095 TERMINATED BY USER REQUEST
```

If you press attention in response to a terminal input request, and then elect to continue (by responding with a null line), you are prompted again for the input that was interrupted.

Cancel Output

A cancel output interrupt causes all further output from your program to be suppressed. This status is reset:

- when terminal input is required, either for an INPUT or a PAUSE statement (you will receive the '?' prompt)
- when you press attention

Processing Time Limit

When your processing time limit is reached, VS BASIC sends you the message:

```
ICD094 LINE xxxxx MAXIMUM CPU LIMIT EXCEEDED
```

You are then prompted for input from your terminal. If you respond with a null line (carrier return only), execution continues. If you enter any data or press attention, execution is terminated and a message is sent to the terminal.

```
ICD095 TERMINATED BY USER REQUEST
```

Note: Message ICD094 is sent immediately before any output queued by VS BASIC is sent to the terminal.

Session Termination

Execution of your program is terminated immediately.

Chaining Programs

With the program chaining technique, VS BASIC programs can be shared with other VS BASIC programs and with other users. See *VS BASIC Language* for details of BASIC rules. For VSPC you need to know two things:

1. The first parameter (the chained program) in CHAIN must be a valid VSPC filename.
2. A nonnumeric second parameter can be used to pass information to the chained program, if the program to which you are chaining is VS BASIC or VSPC FORTRAN. As many as 255 characters of information can be passed.

The CHAIN statement is used in one program to tell the computer to terminate the current program and start executing another program. To tell the computer which program to start executing, you name it in the CHAIN statement. For example,

```
500 CHAIN 'PROGB', A$
```

where A$ is a character variable not more than 255 characters long.

This statement tells the computer to begin executing the program named PROGB, using information contained in A$. Note that when the CHAIN statement is executed, the current program (the program containing the CHAIN statement) is terminated.

Command List (CLIST) Processing

A command list is a series of VSPC commands entered into your workspace as data. VSPC commands are immediately executable, but a command list is not—it is first created and then executed later as a group of commands.

Once a command list is created, it can be executed immediately, and/or it can be saved in a library for future reference. The sequence of commands can be executed using the RUN command.

When you create a command list, you enter the series of commands, preceding each one with a line number. You can either type the line numbers yourself, or you can use INPUT mode to generate them automatically.

Once the entire list is completed, you must make sure the workspace attribute is CLIST before you process the list. Then you can execute the list immediately using the RUN command, and/or you can place it in a library using the SAVE command.

You can use command lists to execute a series of commands you know you'll always use whenever you're doing a specific type of processing.

For example, you could execute the following command list whenever you know you're going to create long precision LBASIC programs:

```
00010 TRANSLATE CAPS
00020 CLEAR
00030 ENTER LBASIC
```

You may want to run a program that requires a temporary file. With CLIST you can create and purge the file automatically.

```
00010 FILE TEMP
00020 RUN SORT
00030 PURGE TEMP
```

In either case, you'll be able to use one RUN command to execute all of the commands in the list.

During execution (with one exception) each command is displayed at the terminal just before its execution in the following form:

```
*line-number command
```

The exception is that, if the command list has the NOREAD characteristic, the commands are not displayed.

In either case, however, output is displayed at the terminal. Such output can be from an executing command (for example, a FIND command) or from a program (for example, a VS BASIC program invoked by a RUN command).

As soon as one command has been processed, the next command in the list is executed. After the last command in the list has been processed, the terminal keyboard unlocks for further terminal input. When a CLIST contains a RUN command naming a second CLIST, return is made to the next command in the first CLIST following completion of the second CLIST.

USING FILES

Several VSPC commands help you to manipulate your files. The FILE command creates or changes characteristics of a file. PURGE eliminates a file. PROTECT and SHARE define the availability of your files to other users.

Creating and Changing a File

You can create data using either VS BASIC, or you can create your data using VSPC and the DATA attribute. Data files are initially defined through the VSPC FILE command.

FILE Command

The FILE command creates a new file in a library or alters the characteristics of an existing file. FILE can set or change the maximum size and it can specify or change the type of file organization.

FILE	*filename*	Required first operand.
	n	Optional second operand. If omitted for new files, editable file size limit is assumed; for existing files present file size limit is assumed.
	SEQUENTIAL **DIRECT**	Optional; choose one or omit. If omitted, SEQUENTIAL is assumed for new files; for existing files, the present organization is assumed.
	UNDEFINED	Optional; specify or omit. If omitted, file is editable.
	CONTENT*(attribute)*	Optional; choose one or omit. If omitted, attribute is DATA for new files; unchanged for existing files.

filename
identifies the file to be created or modified. The name must conform to the rules for VSPC filenames (see "Filenames" section). The CONTINUE file must not be specified.

n
specifies in thousands of bytes the maximum size permitted for this file. It must be an integer from 0 through 65535.

SEQUENTIAL
identifies this file as a sequentially organized file.

DIRECT
identifies this file as a direct file.

UNDEFINED
identifies this file as a foreground processor data file (for example, one created by a VS BASIC program) that cannot be edited by VSPC. The file organization can be sequential or direct; if neither the SEQUENTIAL nor DIRECT operands is specified, SEQUENTIAL is assumed.

CONTENT*(attribute)*
specifies attribute for the resulting file: a foreground processor, CLIST, or DATA. If CONTENT is omitted for new files, the DATA attribute is assumed. If CONTENT is omitted for existing files, the content attribute is unchanged.

File Creation

If you are creating a new file, n specifies the maximum size. If n is omitted, the file limit for editable files specified in your profile is assumed. In this case, when the FILE command is executed, a directory entry with the specified attribute is created, but no space is allocated to it.

File Modification

At least one operand in addition to *filename* must be specified. When the file already exists, the following warning message is sent:

```
WARNING - FILE ALREADY EXISTS
```

and the FILE command is executed.

In this case, *n* specifies the new maximum size allowed for this file. If *n* is omitted, the present maximum size is unchanged. If *n* is specified, and the file is larger than *n,* the following warning message replaces the previous warning message:

```
WARNING - FILE LIMIT LESS THAN EXISTING FILE SIZE
```

The limit will then be set to the value of *n*. The size of present file can then no longer be increased, and, if it is ever rewritten, *n* will become the new maximum file size. (Note, therefore, that zero (0) is a legal maximum file size specification.)

The SEQUENTIAL, DIRECT, and/or UNDEFINED operand respecifies the file organization. If the operand is omitted, the present file organization is unchanged.

If DIRECT is specified, and the existing file does not meet the requirements for a direct file, the command is rejected.

A nonempty sequential or direct file cannot be changed to an undefined sequential or direct file, or vice-versa. An object file cannot be changed to any other type.

Examples:

```
file 456789 sunny/pass 40 direct
```

creates a file directory entry called 'sunny' with a password PASS; the file has a size limit of 40,000 bytes and DIRECT organization; the directory entry is for project library 456789.

```
fil road
```

creates a file directory entry called ROAD with the SEQUENTIAL organization default; the directory entry is created for your own library.

```
file rain 10 CONTENT(BASIC)
```

creates a file directory entry called 'rain' with no password; the file has a size limit of 10,000 bytes and the attribute BASIC.

Using VS BASIC to Create Data

You can create data using a PUT statement or a WRITE FILE statement. Later, in the same program or any other program, you can retrieve the file using a GET statement or READ FILE statement. This is true until the file is purged.

Note that if you create data using the PUT statement, you must retrieve it with a GET statement; if you create it with a WRITE FILE statement, you must retrieve it with a READ FILE statement.

You can also write data to a file by using the [MAT] PRINT [USING] statement in conjunction with the PRINT TO statement. The PRINT TO statement is especially useful in obtaining permanent, hard copy of a report.

Using VSPC to Create Data

You can use the DATA attribute in VSPC to create test data. The ENTER DATA command gives you the DATA attribute. You then type all the test data you need. You can use the NAME command to identify the test data and a SAVE command to place it in a library. The test data can then be retrieved with a VS BASIC GET statement or READ FILE.

PURGE Command

The PURGE command removes a file from a library.

PURGE	*filename*	Required.

filename
identifies the file to be removed. It must conform to the rules for a filename (see "Filenames" section).

If the file has a password, the password can be omitted when the PURGE command is issued. If, however, the password is specified, it must be specified correctly, or the command is rejected.

In any library, the file owner can successfully issue a PURGE command. For files in public or project libraries, the library manager may also specify the PURGE command; for other users, this command is rejected.

If PROTECT NOWRITE is in effect for this file, the PURGE command is rejected. See "Controlling Library Access."

Examples:

```
purge 456789 rainy/weather
```

deletes RAINY from project library 456789.

```
pur 0 rainy
```

deletes RAINY from project library 456789. (Note that the password has been omitted.)

Controlling Library Access

In addition to the security provided by the two forms of the use of passwords (logon and filename), VSPC provides the PROTECT and SHARE commands that modify the file access rules.

SHARE Command

The SHARE command controls access to your files by other users.

SHARE	*filename* *	Required first operand; choose one.
	YES **NO**	Optional; choose one or omit. If omitted, YES is assumed.

filename
specifies the file with which you want to associate new characteristics; *filename* must conform to the rules for VSPC filenames (see "Filenames" section).

*
indicates that the SHARE characteristic specified is to be associated with the current contents of your workspace. The characteristic takes effect when the workspace is placed in a library.

YES
specifies that the file can be accessed by all VSPC users.

NO
specifies that the file can be accessed only by users with normal access to this library.

If you specify SHARE filename with no other operand (YES or NO), then YES takes effect. However, if you have not specified a SHARE command at all, then SHARE NO is in effect.

After SHARE NO is specified for a private library file, only the owner can access the file. After SHARE NO is specified for a project library file, only users of that project library can access the file.

The SHARE command does not affect access to a public library file and cannot be issued for the special file CONTINUE.

You can issue the SHARE command only if you are the owner of the file.

After a SHARE command is issued, it remains in effect until another SHARE command changes the SHARE characteristic.

Examples:

```
sha *
```

After you place your workspace in a library with a SAVE command, the resulting file can be shared with other users.

```
share 1234567 random/cat no
```

The file RANDOM in your own library can no longer be accessed by other users.

PROTECT Command

The PROTECT command changes access to a currently existing file or to one that may be created later.

PROTECT	*filename* *	Required first operand; choose one.
	PASSWORD **PASSWORD(***password***)** **NOPASS**	Optional; choose one or omit. If omitted, password protection unchanged.
	READ **NOREAD**	Optional; choose one or omit. If omitted, access characteristic unchanged.
	WRITE **NOWRITE**	Optional; choose one or omit. If omitted, access characteristic unchanged.

filename
specifies the file to which the command applies; *filename* must conform to the VSPC rules for filenames (see "Filenames" section).

*
indicates that the PROTECT characteristic specified is to be associated with the current contents of your workspace. The characteristic takes effect when the workspace is placed in a library.

PASSWORD
indicates you want to be prompted to enter a new password (see "Passwords" section).

PASSWORD *(password)*
applies a password to the specified file (see "Passwords" section).

NOPASS
removes an existing password.

READ
allows users other than the owner to use the file as input for an executing program or to include it in a SUBMIT job entry command. READ also changes a previous NOREAD characteristic.

NOREAD
prevents users other than the owner from using the file as input for an executing program, or including it in a SUBMIT job entry command. It also changes a previous READ characteristic. Another user may, however, specify this file in a LOAD, MERGE, or RUN command; if he has access to the file. The NOREAD operand may not be specified for an object program file.

When a LOAD, RUN, or MERGE command places a file with NOREAD characteristics in the workspace, the entire workspace assumes the protected characteristic. This means that the following commands are rejected: SAVE, STORE, CHANGE, FIND, and LIST.

WRITE
specifies that this file's owner may modify this file.

NOWRITE
protects the file from modification, even by its owner. The file cannot be opened for output or update by an executing program, nor can it be named in a SAVE, STORE, or PURGE command.

Only the file's owner may specify the PROTECT command. When you issue the PROTECT command, you must specify at least one of the optional operands.

The PROTECT command cannot be issued for the special library member CONTINUE.

Newly created library members have the READ and WRITE characteristics.

If WRITE is the only keyword operand, and a password for the file exists, the password can be omitted. However, if the password is specified, it must be specified correctly or the command is rejected.

Examples:

These examples assume you have issued the SHARE command on the named file.

```
protect 1234567 random/cat password(pass)
read nowrite
```

protects the indicated file with the new password, PASS, and indicates that another user can read, run, or list it, but that you yourself cannot change it.

```
pro program2 nor
```

protects the named file from being accessed by users other than the owner (except with the RUN command).

RELEASE Command

The RELEASE command allows another user to use the ACQUIRE command to gain ownership of your noncontrolled project file.

RELEASE	*filename*	Required.

filename
specifies the name of the file to be made available for transfer.

The owner of a file makes it eligible for transfer with the RELEASE command. If must be a noncontrolled project library file. The file is still owned by the original user until another user issues an ACQUIRE command naming the file.

A file belonging to the library manager is not eligible to be transferred.

ACQUIRE Command

The ACQUIRE command transfers ownership of a noncontrolled project file.

ACQUIRE	*filename*	Required.

filename
specifies the name of the file to be transferred. The file must be in a noncontrolled project library.

When you issue the ACQUIRE command, VSPC checks that the file is in a noncontrolled project library, that you have enough space to store the file, and that the current owner has issued the RELEASE command.

If all these conditions are met, ownership transfers to you. If not, the command is rejected and transfer does not take place.

The library manager of the project library cannot use the ACQUIRE command to become the owner of files owned by other project users.

Note: The current owner of a file can use the ACQUIRE command to retract the result of a previous RELEASE command.

LIMITS AND CONTROLS

Two major items involved in your work at the terminal have limits: the length of your line and program size.

Line Length

The length of the line at your terminal is determined by the type of terminal you have. You can type characters, backspaces, tabs, in any order and as many as you like, until you reach the right-hand margin.

In the compiler a different set of rules applies. For example, a backspace is considered a character in the compiler. On your terminal you can type across the carriage, backspace to the left margin, type across again, and repeat this as often as you like *mechanically.* But, if you try to send that line to the compiler, it is rejected with an error message.

The maximum line length of your source program that the compiler can handle is 247 characters, excluding the line number. If a line in your program exceeds this length, VS BASIC prints an error message when you try to run your program.

The maximum terminal input line length during program execution is 255 characters. That is, when you are running a program at the terminal and an input line is requested, you cannot enter more than 255 characters or an error message will print.

Program Size

Two "containers" are involved in producing the results of your program: your workspace has a defined size and the compiler (the computer program that translates the program in your workspace into a language the machine can read) is limited in the size of the program it can handle.

Your workspace has a limit established by your VSPC administrator (maximum is 510,000 bytes). If your program exceeds this limit, a message prints at your terminal during compilation.

The maximum source program that the VS BASIC compiler can handle is 65,537 bytes; the highest VSPC line number is 99999; the maximum number of VS BASIC source program statements is 1000. If you attempt to run a program exceeding these limits, you will get an error message.

The REM clause can be used instead of the REM statement to reduce the number of source statements needed to document a VS BASIC program.

FILE ORGANIZATION

The files in a VSPC library may contain five types of files: sequential, direct, object program, undefined sequential, and undefined direct. In addition, you can access VSAM files stored outside the VSPC library.

VS BASIC processes VSPC files using the record or stream I/O statements. A full description of each such statement is given in *VS BASIC Language.*

VSPC Sequential Files

A VSPC sequential file is one in which the records are positioned in the order they are created. Records can then be retrieved by VS BASIC programs in the order they were created. Records in the file can be of varying lengths.

A VSPC sequential file must consist of editable data; that is, the file must have one of the following attributes: DATA, CLIST, BASIC (source statements), or one of the other VSPC processors. Each record must begin with a line number. When the file is loaded into your workspace, records in the file can be edited (that is, modified, added, or deleted) in any line number order.

VSPC Sequential DATA Files

VSPC sequential files with the DATA attribute may be processed by either VS BASIC entry-sequenced record I/O statements or VS BASIC stream I/O statements. (VS BASIC does not allow the use of the KEY or the REC clause in I/O statements used on sequential files. If a user attempts to do this, the program terminates with an error message.)

Stream I/O Statements

VS BASIC creates stream data with the PUT statements in records of the current terminal linesize minus six; the GET statement retrieves such files. Only the data portion of each VSPC record can be read and written and will have a minimum length of 30 and a maximum length of 249. The files are written with a line number starting at one and incrementing by one.

Entry-Sequenced Record I/O Statements

All entry-sequenced record I/O statements are supported. Only the data portion of each VSPC record can be read and written by the program. The maximum length of each record is 4058 bytes. The line numbers of each record start at one and increment by one.

You use WRITE FILE to add a new record. To change an existing record, you use REWRITE FILE. You can retrieve records with READ FILE and REREAD FILE statements. You can reposition a file using the RESET FILE statement.

For entry-sequenced data, you can also transmit to a record-oriented file by using the [MAT] PRINT [USING] statement in conjunction with the PRINT TO statement.

You can retrieve data using the [MAT] INPUT statement in conjunction with the INPUT FROM statement. Data retrieved using this method must be on a record-oriented file.

Although you cannot use READ FILE or WRITE FILE with either INPUT FROM or PRINT TO, you can use RESET FILE, OPEN FILE, and CLOSE FILE.

The REUSE clause can be specified in an OPEN statement to reuse an existing file. A file opened with REUSE can thus be used as a work file.

VSPC Sequential Non-DATA Files

VSPC sequential files with the non-DATA attribute are only supported by the VS BASIC entry-sequenced record I/O statements (WRITE FILE, REWRITE FILE, READ FILE, RESET FILE, and REREAD FILE or [MAT] PRINT [USING] in conjunction with PRINT TO and [MAT] INPUT in conjunction with INPUT FROM). Any attempt to use stream I/O statements will result in the program terminating with an error message.

When a record is retrieved with the READ FILE or REREAD FILE statements, the line number is converted to a five-digit character string followed by a space and with the data portion of the record appended to it.

When a record is retrieved with INPUT associated with INPUT FROM, the line number is converted and passed to the first variable in the I/O input list. Therefore, the first variable in the list must be a numeric variable and is used by VS BASIC to store this line number.

When a record is written (with a WRITE FILE statement) or rewritten (with a REWRITE FILE statement), the beginning of the record must contain a five-digit line number followed by a blank. Any leading blanks are treated as zeros but an all-blank line number is invalid. A line number greater than 99999 is invalid. A line number in a rewritten line that is not the same as the one read is invalid.

The maximum record size is 4064 bytes.

If the line number is invalid, the program terminates with an error message.

The line numbers must be written in ascending sequence; otherwise, the program terminates with an error message.

VSPC Direct Files

You can read records from and write records to VSPC direct files by using the relative-record capabilities of VS BASIC. You use these capabilities by specifying the REC clause in the VS BASIC I/O statements for record-oriented files. The REC clause is discussed in the publication *VS BASIC Language.* When using relative-record VSAM files, you can read records either sequentially or by their relative-record number, but you can write records only directly, by specifying their relative-record number.

A VSPC direct file must have all the following specifications:

- The file attribute must be DATA.
- Line numbers must be present and must begin at one and increment by one.
- All records must be the same length.

Object Program Files

You create an object program file with the STORE command which compiles a source program. Such a file consists of machine language object program instructions. Object program files are made available to you through the RUN command (see "Compiling and Executing a Program"). They cannot be edited.

VSPC Undefined Files

Access to undefined files is through stream I/O statements. On input, the data is checked for compatibility with the target variable and converted if necessary. If the data type is unknown or its value is invalid, the program is terminated with an error message.

- Conversion from an unsupported data type on the input file is not allowed.
- A numeric data item cannot be input into a character variable.
- A character data item cannot be input into an arithmetic variable.
- Output to a direct undefined file is not allowed.

Subject to the restrictions stated, undefined files written by VSPC FORTRAN can be read by VS BASIC. (See Figure 12.)

Data Type	Length	Read	Write	Valid Target Data Types
Integer	2	x		
	4	x		
	8	x		
				arithmetic variable
Float	4	x	x	
	8	x	x	
	16	x		
Complex	8	x		arithmetic variable
	16	x		(real part only)
	32	x		
Logical	1			none
	4			
Character		x	x	character variable

Figure 12. Undefined File Conversion

VSPC External VSAM Files

In some situations, external files (that is, files with VSAM organization stored outside the VSPC library) are available directly to VS BASIC through one specific library number. You access such files through that library number, which is determined by your VSPC administrator. External VSAM files cannot be defined using VSPC commands.

The record-oriented file statements of VS BASIC are used to process external VSAM files. These files may be entry-sequenced, key-sequenced, or relative-record VSAM files. Record processing of external VSAM files may be done via primary or alternate indexes.

Check with your system administrator whether or not external VSAM files have been defined for your use.

Figure 13 illustrates the relationship between VS BASIC input/output statements and VS BASIC files.

File Access Statements

Input	Output	Access Type	File Organization	VSAM Type
GET	PUT	Sequential	Stream	
READ FILE	WRITE FILE	Sequential	Record	ESDS
(INPUT FROM) INPUT	(PRINT TO) PRINT	Sequential	Record	ESDS
READ FILE KEY= Note 1	WRITE FILE Note 2	Direct	Record	KSDS
READ FILE REC= Note 3	WRITE FILE REC=	Direct	Record	RRDS

Notes:

1. Subsequent READ FILE without a KEY clause reads next record in key sequence.
2. Key is part of the record; cannot be specified with a key clause.
3. Subsequent READ FILE without a REC clause returns next sequential non-null record.

Figure 13. Relationship between Input/Output Statements and Files

CREATING AND USING VS BASIC STREAM FILES

Before you create a stream I/O file with a PUT statement in a VS BASIC program, it must be defined by a VSPC FILE command. The VSPC filename is the name you use in the PUT statement. So, if you, as usernum 1234567, have a PUT statement for a file previously defined as NEWFILE and with a password, PASS, then you must specify the PUT statement as

```
PUT '1234567 NEWFILE/PASS',A,B
```

or

```
PUT 'NEWFILE/PASS',A,B
```

Creating a Stream File Through VSPC

You can create a stream I/O file as a data file through VSPC. You do this as if you were entering lines in a VS BASIC program, only instead of working with the BASIC or LBASIC attribute, you specify ENTER DATA. To create a file named STREAM, for example, you first enter the following commands:

```
clear
enter data
name stream
input
```

After you issue these commands (you could create a command list if you're going to use them often), type the contents of your file. Character data must be enclosed in quotation marks, and all data items (numeric or character) must be separated from each other by a single comma or by one or more blanks.

You can type a stream file called ADDR that keeps each employee's address:

```
clear
READY
enter data
READY
name addr
READY
input
00010 'j.h. adams' '700 lincoln' 'boulder co.'
00020 'b.a. berry' '14 el camino real' 'sunnyvale ca.'
00030 't. clark' '2589 marquette' 'minneapolis, mn.'
00040
READY
```

When you have finished, issue a SAVE command. You have now created a data file that can be edited, listed, saved, and renamed. The name attached to this file is 1234567 ADDR.

You can now access this file in a GET statement in any VS BASIC program:

```
60 get 'addr'
```

File Limits

The n operand of your FILE command determines when you reach the end of the files you're creating through the VS BASIC program. The maximum number of files that can be open at the same time is 15.

File Security

The VSPC library manipulation routines provide a comprehensive set of facilities to both share data files and to keep them secure. These facilities are described in the section "Controlling Library Access."

If you specify OPEN ALL HOLD in your source program, no other user can access your file while you're updating a record. That is, your file is locked at the record level and not at the file level. (See *VS BASIC Language.*)

During execution, if your VS BASIC program attempts to open a shared file that is locked for output, it will retry once after a delay of about 15 seconds and, if the file is still locked, the program will terminate with an error message.

The protection characteristics of the file are checked when the file is opened. If the protection characteristic does not allow access to the file, the program will terminate with an error message.

A VS BASIC program will terminate with an error message if you attempt:

- to open another user's file for IN or ALL that has PROTECT NOREAD in effect
- to open one of your own files for OUT or ALL that has PROTECT NOWRITE in effect
- to update a file you don't own
- to write a file larger than the file size specified for the file. This size may be by default (grown too large to be editable), or it may be specified with the FILE command.
- to process a file larger than your library can handle
- to open an OBJECT file
- to open an undefined file for record-oriented I/O processing.

CROSS-LANGUAGE DATA EXCHANGE USING VSPC FILES

Your VS BASIC programs can use files other than those created by you in VS BASIC programs. These files might be:

- VS BASIC files created by another user.
- Files created by VSPC FORTRAN or VS APL. (These files are readable only under certain conditions.)

Exchanging Data with VSPC FORTRAN

Data formats vary between VS BASIC and VSPC FORTRAN. Certain numeric data is compatible except for differences in precision. Both record-oriented and stream files can be exchanged but with some modifications.

VS BASIC Record-Oriented Files and VSPC FORTRAN Files

VS BASIC record-oriented files can be read by VSPC FORTRAN formatted and unformatted record I/O statements. Similarly, VSPC FORTRAN files written by formatted or unformatted I/O statements can be read by VS BASIC record-oriented I/O statements. However, you must be sure that the data types match.

VS BASIC Stream Files and VSPC FORTRAN Data Files

The VS BASIC stream I/O facility is similar in many ways to VSPC FORTRAN's list-directed I/O. However, VS BASIC does not recognize record boundaries except where they are necessary for writing the file. VS BASIC can read a VSPC FORTRAN list-directed data file record, provided that neither COMPLEX nor LOGICAL FORTRAN data types are used. Note that VS BASIC writes arrays in row major order and VSPC FORTRAN in column major order. Therefore, they must be transposed before they can be used by any FORTRAN array operator.

VS BASIC Stream Files and VSPC FORTRAN Undefined Files

The VS BASIC stream I/O facility is similar in many ways to VSPC FORTRAN's list-directed I/O. However, VS BASIC does not recognize record boundaries except where they are necessary for writing the file. VS BASIC can read a VSPC FORTRAN list-directed data file record, provided that LOGICAL FORTRAN data types are not used. Note that VS BASIC writes arrays in row major order and VSPC FORTRAN in column major order. Therefore, they must be transposed before they can be used by any FORTRAN array operator.

Exchanging Data with VS APL

VS APL's EBCDIC files can be read by VS BASIC I/O statements. For stream GET statements, the VS APL file must be written so that fields are separated by blanks. For record-oriented I/O statements, each character vector written by VS APL will appear as a separate record.

VS BASIC's stream files can be read by VS APL, but the character vector read must be decoded by the receiving function. For files containing only numeric data, the EXECUTE operator may be used. Each record of a VS BASIC VSPC file can be read into a character vector. Each item in the record should be in EBCDIC if it is to be interpreted by VS APL and the receiving function must decode it.

MESSAGES FROM THE SYSTEMS

All messages have identifiers (consisting of a series of letters and numbers) preceding them. The letters tell you what part of the system has produced the message. The prefix letters for VSPC are ASU; the prefix letters for messages produced by the VS BASIC processor are ICD. These identifiers are not printed at the terminal unless you have entered the MESSAGE ID command.

VSPC Error Messages

When VSPC finds an error in a command, it will issue an error message and you will be asked to retype the entire command. For example:

```
load 123
ASU203 FILENAME ERROR
```

If there are errors in the operation of VSPC itself, you may get one of the following VSPC messages:

```
ASU200 SYSTEM PROBLEM 0
ASU207 SYSTEM PROBLEM n
```

where n is a nonzero number. Save this number as information for your system administrator.

Explanations of VSPC terminal messages are listed in a computer printout that can be obtained by the procedure described in the *VSPC General User's Guide and Command Language.* These explanations can also be displayed at your terminal using VSPC AID; enter `?ASU203` for an explanation of message ASU203, or `?n` for an explanation of the *n*th preceding VSPC message.

VS BASIC Error Messages

VS BASIC messages are issued for errors in three categories:

1. Syntax errors: errors in the structure of a statement (for example, erroneous punctuation).
2. Semantic errors: errors in the structure of your program (for example, invalid nesting, invalid combination of data types, missing end statements, and so on).
3. Execution errors: errors detected during the execution of the program (for example, dividing by zero, differences in the attributes of arguments and operands, improper subscript values, and so on.)

Each VS BASIC message is self-explanatory. It consists of text preceded by a six-character identification code (ICD plus a three-digit number) and, in some cases, a line number. For example:

```
ICD210 LINE 70: EXPRESSION REQUIRED AT X=
```

VS BASIC error messages with diagnostic aids are distributed as part of the installation procedure. Check with your VSPC administrator.

If you are using the ON statement with any of its error clauses, some of the errors normally encountered by VS BASIC and indicated by error messages are trapped and returned to your program. The error is relayed to the program in the form of a return code which corresponds to the message number that would have been printed.

SAMPLE SESSION

Figure 14 contains a hypothetical terminal session and provides a description of the session. The session illustrates many VSPC features as well as the interaction that takes place between you, VSPC, and the VS BASIC processor in a typical application.

Significant points in the session are indicated by numbers in the left margin; each number has a corresponding explanation at the end of the printed session.

An IBM 2741 Communications Terminal was used for the session. The times (TIME) depend on the level of system activity, the tuning and performance of VSPC, and the CPU model.

```
    vspc id=1234567 @
    ENTER PASSWORD
1   ■■■■■■
    BASIC 09/04/78 09:16:20 1234567
    WELCOME TO VSPC
    READY
    name commdiv/pass
    READY
2   input
    00010 rem this program finds the greatest common divisor of
    00020 rem any two short precision positive arithmetic
3   00030 ren
            m integer variables (represented by x and y).
    00040 input x1,y1 rem accept two numbers
    00050 x=max (x1,y1
    00060 y=min (x1,y1)
    00070 if x .le. 0 .or. y .le. 0 then 90 else 100
4   00080 carrier return
    READY
    change 70 '9' '8' text
       70 IF X .LE. 0 .OR. Y .LE. 0 THEN 80 ELSE 100
    1 LINES CHANGED
    save
    READY
    input
    00080 print 'invalid values; try again with new ones.'
    00090 goto 40
    00100 ON ZDIV then 190 rem prepare if division by zero
    00110 if x-y=0 then 160 else 130
    00120 r=x-(int(x/y)*y)
    00130 x=y
    00140 y=r
5   00150 carrier return
    READY
    120 r=x-(int(x/y))*y
6   input
    00150 goto 100
    00160 print using 170,y
    00170 : the common divisor is: ######
    00180 stop
    00190 print 'division by zero has occurred at line'; &line
    00200 end
    00201 carrier return
    READY
```

Figure 14 (Part 1 of 2). Sample Session

```
7    195 stop
     save
     READY
8    list
      10 REM THIS PROGRAM FINDS THE GREATEST COMMON DIVISOR OF
      20 REM ANY TWO SHORT PRECISION POSITIVE ARITHMETIC
      30 REM INTEGER VARIABLES (REPRESENTED BY X AND Y)
      40 INPUT X1,Y1 REM ACCEPT TWO NUMBERS
      50 X = MAX(X1,Y1
      60 Y = MIN(X1,Y1)
      70 IF X .LE. 0 .OR. Y .LE. 0 THEN 80 ELSE 100
      80 PRINT 'INVALID VALUES; TRY AGAIN WITH NEW ONES.'
      90 GO TO 40
     100 ON ZDIV THEN 190 REM PREPARE IF DIVISION BY ZERO OCCURS
     110 IF R - Y = 0 THEN 160 ELSE 130
     120 R = X - (INT(X/Y))*Y
     130 X = Y
     140 Y = R
     150 GO TO 100
     160 PRINT USING 170,Y
     170 : THE COMMON DIVISOR IS: #######
     180 STOP
     190 PRINT 'DIVISION BY ZERO HAS OCCURRED AT LINE'; &LINE
     195 STOP
     200 END
9    run
     COMMDIV 09/04/78 10:20:23
     LINE 50: UNBALANCED PARENTHESES
     TIME 0.5 SECS
10   50 x=max (x1,y1)
     save
     READY
     run
     COMMDIV 09/04/78 10:57:35
11   ?
     9,0
12   INVALID VALUES: TRY AGAIN WITH NEW ONES.
     ?
     9,4
13   DIVISION BY ZERO HAS OCCURRED AT LINE 110
14   110 r=x-(int(x/y))*y
     120 if r=0 then 160 else 130
     save
     READY
     run
     COMMDIV 09/04/78 11:10:10
     ?
     9,4
     THE COMMON DIVISOR IS: 1
     TIME 0.5 SECS
15   store DIVCOM/pass
     DIVCOM 09/04/78 11:17:25
     COMPILED SIZE 10176 BYTES
     TIME 0.5 SECS
     off
     09/04/78 11:15:13 1234567
     CONNECTED  03:33:41
     CPU TIME 1
```

Figure 14 (Part 2 of 2). Sample Session

1. VSPC types and retypes a series of letters so that when your password is entered over these letters (blot), it cannot be read by another user.
2. You enter the INPUT command for automatic line numbering (with prompting).
3. The typing error n for m is corrected with the use of the backspace, ATTN, retype correction facility.
4. You notice a mistake in line 70 so, when line 80 prints, you press carrier return, and enter a CHANGE command with the TEXT option to change THEN 90 to THEN 80. VSPC verifies the change on the following line. You type SAVE; VSPC says READY, and you type another INPUT command. VSPC prints line 80 again, after which you type your next line.
5. You are about to enter a statement at line number 150, but see an error in line 120, so you press carrier return and then replace it by typing the line number and the corrected line.
6. After correction, INPUT is required to resume automatic line numbering (Automatic line numbering resumes with a number 10 greater than the last existing line, in this case, 150.)
7. After you have completed your program, you notice that you omitted a statement after line 190. You insert the line, numbering it 195, and issue another SAVE.
8. You want to see a copy of your program. The LIST command shows you a sorted copy with modifications and additions incorporated.
9. You enter a RUN command to compile and execute your program. The compiler finds an error that prevents execution and types out error message ICD246 (without the identification number because you have not specified MESSAGE ID). There is a missing parenthesis in line 50.
10. You correct line 50, save the program again, and run it again.
11. This question mark is printed by VS BASIC and requests input data for the INPUT statement in line 40. In response, you enter two values (9 and 0).
12. This message is produced by your program as a result of a branch to line 80. The branch occurs because Y was entered as 0; the program does not accept values that are less than or equal to 0. The INPUT statement is then reissued (note the GOTO statement at line 90), and the request for input data is printed again. Again, you type two values.
13. This time a message is produced indicating that division by zero is taking place in line 110.
14. In reviewing your program to find the error, you determine that R should be set to zero only after the common divisor is found. Rather than test for x-y=0, the program should test for r=0. Statements 100 and 110 need correction. You replace statements 110 and 120 with the statements you think are needed. You save the updated program, and re-execute it, providing the input values when requested. This time the program works.
15. Satisfied with your results, you issue a STORE command to compile your program and send it to your library. VSPC responds and you issue an OFF command to end your session. VSPC prints out some final informational messages.

APPENDIX A. DEVICE DEPENDENCIES

In addition to the commands already described, special VSPC commands can be used with the CPT-TWX terminals.

CPT-TWX Terminal Special Commands

Three commands are unique to the CPT-TWX terminal:

- KEY Command to determine whether or not the keyboard is the source of input.
- TAPE Command to determine whether or not paper tape is the source of input.
- PUNCH Command to determine whether or not the output is to be punched on paper tape.

The LINESIZE command has an operand (x) used only with CPT-TWX terminals to specify the number of idle characters to be inserted following each carrier return.

KEY Command

The KEY command specifies that the CPT-TWX keyboard is the source for terminal input.

KEY		No operands allowed.

At logon, the keyboard is the assumed input source. Whether or not terminal input is read from the keyboard depends on the manual setup of the CPT-TWX terminal. If the manual setup is wrong, input is incorrectly formatted on the console sheet.

When the KEY command is in effect, VSPC automatically causes a line feed after each input line is entered.

Example:

```
key
```

tells VSPC you are using the keyboard on the CPT-TWX terminal.

PUNCH Command

The PUNCH command specifies whether or not output is punched on paper tape.

PUNCH	**ON** **OFF** **CONTROL**	Optional; choose one or omit. If omitted, ON is assumed.

ON
terminal output will be punched on paper tape.

OFF
terminal output will not be punched on paper tape.

CONTROL
specifies that all possible ASCII codes can be sent to the terminal without modification by VSPC.

Whether or not terminal output is punched on paper tape depends on the manual setup of the CPT-TWX terminal.

When PUNCH ON is in effect, VSPC adds the characters

```
XOFF RUBOUT RUBOUT RUBOUT
```

at the end of each output line and no line feed is performed.

When PUNCH OFF is in effect, no characters are added at the end of a line. PUNCH OFF is assumed at logon.

Whether or not terminal output is punched on paper tape depends on the manual setup of the terminal. If the setup is wrong, the punched output is either incorrectly formatted or nonexistent.

Examples:

```
punch off
```

no characters are added at the end of the line.

```
pun on
```

VSPC adds characters XOFF RUBOUT RUBOUT RUBOUT to the end of each line of output.

TAPE Command

The TAPE command specifies that the paper tape reader is the source for terminal input.

TAPE		No operands allowed.

When the TAPE command is in effect, VSPC does not cause a line feed after each input line on the keyboard.

Before each input line VSPC sends an X-ON character to the terminal.

Whether or not terminal input is read from tape depends on the manual setup of the CPT-TWX terminal. If the setup is wrong and TAPE is specified, lines may be overprinted.

Examples:

```
tape
```

you will use paper tape for your input on the CPT-TWX terminal.

```
tap
```

you will use paper tape for your input on the CPT-TWX terminal.

APPENDIX B. CONVERSION TO VS BASIC UNDER VSPC

VS BASIC source programs and data files can be migrated to and from the VSPC library by the VSPC Service Programs.

Migration to VSPC from CALL-OS and ITF

ITF BASIC source program and data files and CALL-OS BASIC may be copied into the VSPC library by the VSPC Service Program after they have been placed in normal OS data sets. For CALL-OS source program and data files, the CALL-OS utility program DIBCADBU can be used to convert the files to normal OS data sets (using the FORMDATA option, in the case of data files).

All correctly running ITF BASIC source programs will be run correctly by VS BASIC under VSPC after file references are changed to conform to the VSPC file naming conventions. (A program with a large literal pool may require modification to reduce the size of the pool.) Under TSO, ITF BASIC stream data files are written as members of a partitioned data set, where the filename specified in the source program is the member-name of the PDS 'userid.DATA.' Stream I/O statements in VS BASIC under VSPC write to VSPC record-oriented files and specify the file by a character string representation of the entire VSPC filename.

CALL-OS BASIC V1.2 (under CALL-OS and CMS) programs may require modifications to file I/O access and control statements to be run correctly by VS BASIC. The I/O statements OPEN, CLOSE, RESET, GET, PUT in CALL-OS BASIC programs must be changed to replace specification of files by file numbers to specification by character string representation of file name. Under VS BASIC an OPEN to an already open file is ignored. CALL-OS BASIC source programs must be modified in order for an OPEN to a currently open file to be effective. A CLOSE statement should be inserted before the second OPEN. These changes can be accomplished using either the CALL-OS, CMS, or VSPC edit facility.

Source Programs

VSPC provides for the conversion of VSPC files containing VS BASIC source programs to and from fixed- or variable-format sequential data sets (and/or partitioned data set members, in the OS case). See *VS BASIC Installation Reference Material.*

VS BASIC source programs will require changes only to the file references in I/O statements and to the program names in CHAIN statements in order to run correctly.

Source program changes may be accomplished with the VSPC editing facilities.

Data Files

Data file support for VS BASIC is provided by VSPC.

Individual VSAM data sets outside the VSPC library environment are accessible to application programs through the VSPC external VSAM interface. VSAM data sets may be used to provide sharing of data among many applications, in the batch environment as well as under VSPC; the applications may be written in a variety of languages. All VSAM data sets must have been previously defined by using VSAM Access Method Services in the batch environment, and must be included in the Job Control Language deck for VSPC startup.

The VSPC Service Program will convert VSPC sequential and direct files with the DATA attribute to and from sequential and direct-access data sets (and/or partitioned data set members, in the OS case). VSPC sequential and direct files can be converted to fixed- or variable-format sequential data sets; VSPC direct files can be converted to fixed-format direct-access (BDAM) data sets. Fixed- and variable-format sequential and direct-access data sets can be converted to VSPC sequential files; fixed-format sequential and direct-access data sets can be converted to VSPC direct files.

APPENDIX C. BATCH PROCESSING VPSC JOB ENTRY COMMANDS

The VSPC command language provides a simple set of commands that allow you to submit jobs from your terminal to your computing center for conventional batch processing. While such jobs are being executed, you can use your terminal for your interactive time-sharing applications.

Three requirements must be met for batch processing:

1. The Remote Job Control option must be installed with your VSPC system.
2. Your profile must allow you to use Remote Job Control commands.
3. The VSPC operator must be allowing Remote Job Control at the time you submit your job.

See *VS Personal Computing (VSPC) General User's Guide and Command Language* for details about the job entry commands necessary to use batch processing.

Action	TERMINALS						CPT-TWX Mod 33/35
	3270	2741*	3767*		3770	1050*	
Set switches for computer connection	Pull out OFF/PUSH Push the top (I) of the Power switch.	COM/LCL=COM ON/OFF=ON	Keylock=ON COMM/LOCAL=COMM AUTO/OFF=AUTO (SDLC only) EDIT/OFF=OFF (SDLC only) AUTO VIEW/OFF=as desired DOUBLE/SINGLE SPACE=as desired DATA/TALK=ask administrator DIAL/DISC=ask administrator SDLC/SS=ask administrator Primary/Secondary=as desired CALC/OFF=OFF TEST/OFF=OFF POWER/OFF=POWER PRESS SYS REQ (SDLC)		Keylock=ON POWER=ON HOLDPRINT=OFF EXTENT/ALARM=OFF PUNCH=OFF DISK=OFF UPDATE/MONITOR=OFF INTRP=OFF HALF SPEED=OFF or NORMAL SDLC/BSC=SDLC AUTO=ON (CODE/1 if 3771 or 3773) Press SYS REQ	MAIN-LINE=ON SYSTEM=ATTEND MASTER=ON PRINTER 1=SEND/REC PRINTER 2=HOME KEYBOARD=SEND PUNCH=NORMAL SYSTEM=PROGRAM EOB=AUTO SYSTEM=UP All others=OFF	Press ORIG button Model 35: Press K button On high-pitched tone, press CTRL, WRU
Terminal ready to accept logon	SYSTEM AVAILABLE Cursor appears	Keyboard unlocks	DATA SET READY, PROCEED lights on ONLINE (leased line)		PROCEED, KBD, LINE, and AUTO lights on	POWER, PROCEED lights on	Key mode: paper advance optional? printed
			S/S	SDLC			
Logon: VSPC ID=	usernum	usernum x	usernum x	usernum	usernum	usernum x	usernum
End a line	ENTER	RETURN	↵(return)	↵(return)	↵(return)	RETURN ALTN CODING + EOT	RETURN
Reset terminal	RESET	---	RESET	RESET	CODE and RESET READY Line number	REQUEST	BRK-RLS Model 35: K
Terminal ready to accept input	READY Line number Cursor appears SYSTEM READY	READY Line number Unlocks keyboard	READY Line number ON LINE and PROCEED lights on	READY Line number ON LINE and PROCEED lights on	ON LINE and PROCEED light on ← (backspace) INDEX, retype, ↵(return)	READY Line number Unlocks keyboard PROCEED light on	READY Paper advance Line number Stops noise Bell
Correct current line	Position cursor, ERASE EOF	BKSP, ATTN, retype, RETURN	← (backspace), ATTN, retype, ↵(return)	← (backspace). INDEX, retype, ↵(return)	BUFFER RTN, CNCL	BACKSPACE, ATTENTION, retype. RETURN	Backarrow or underline. retype. RETURN
Delete current line(s)	ERASE INPUT	ATTN	ATTN	BUFFER RTN CNCL		ATTENTION	CTRL/X
Send signal to system	PA1	ATTN	ATTN	ATTN	ATTN	ATTENTION	BREAK
Escape from input	PA1	ATTN	ATTN	ATTN	ATTN	ATTENTION	BREAK
End output	PA2	ATTN	ATTN	CNCL	CNCL	ATTENTION	BREAK
Logoff	OFF	OFF	OFF	OFF	OFF	OFF	OFF
Turn off terminal	Push OFF/PUSH Push the bottom (0) of the Power switch.	ON/OFF=OFF	POWER=OFF Keylock=OFF	POWER=OFF Keylock=OFF	(unnecessary)	MAIN-LINE=OFF	CLR
Default line length	80	120	120	132	132	120	72
Invalid character prints as	" (double quote)	■ (N backspace Z)	■ (N backspace Z)	■ (N backspace Z)	■ (N backspace Z)	■ (N backspace Z)	" (double quote)

*Terminals controlled by the Multiple Terminal Access (MTA) facility of the Network Control Program (NCP) may need special logon procedures. See your VSPC administrator.

APPENDIX E. SUMMARY OF VSPC COMMANDS USED IN VS BASIC

Command	Action
ACQUIRE	Acquires ownership of a noncontrolled PROJECT library file that has been released by another user.
?	(AID) Asks for explanation or prompting.
CHANGE	Changes characters in a line or lines.
CLEAR	Provides an empty workspace.
COPY	Duplicates lines from one location to another in a workspace.
DELETE	Removes one or more lines from your workspace.
ENTER	Specifies the attribute of your workspace.
EXTRACT	Deletes all but specified lines from a workspace.
FILE	Creates or modifies a file in a library.
FIND	Scans a file for a specified character sequence.
HARDCOPY	Used on 3270 temrinal to identify a printer.
INPUT	Begins automatic line numbering.
JOIN	Unites two consecutive lines.
KEY	Used on CPT-TWX terminal to indicate use of keyboard as the input source.
LINESIZE	Sets the length of your output lines.
LIST	Displays workspace contents.
LOAD	Loads a file into your workspace.
LOCATE	Sets current line pointer.
MERGE	Combines a library file with your workspace contents.
MESSAGE	Specifies the way you want messages displayed at your terminal.
MOVE	Rearranges lines in your workspace.
NAME	Assigns a name to the contents of your workspace.
OFF	Ends a terminal session.
PASSWORD	Creates or changes a logon password.
PFKEY	(3270) Requests that a PF key be interpreted as specified.
PROTECT	Specifies the security of a file.
PUNCH	Used on CPT-TWX terminal to indicate use of paper tape punch.
PURGE	Removes a file from a library.
QUERY	Displays current status in VSPC. information about one file, or a list of files in a library.
RELEASE	Releases a file in a noncontrolled PROJECT library so another user can gain ownership of it with ACQUIRE.
RENUMBER	Assigns new line numbers to your workspace contents.
RUN	Executes object program of compiles and executes source program.
SAVE	Puts workspace contents into a library.
SEND	Sends a message from one terminal to another.
SHARE	Allows other users access to library files.
SPLIT	Splits one line into two lines.

STORE	Compiles source program and puts the resulting object program into a library.
TABSET	Communicates terminal tab settings to VSPC.
TAPE	Used on CPT-TWX terminal to indicate use of paper tape reader.
TEXT	Sets global default for the TEXT/NOTEXT operands of the CHANGE, LOCATE, and FIND commands.
TRANSLATE	Specifies use of certain character set.
VIEW	(3270 only) Changes VSPC screen format to VIEW mode (full screen edit) or, if you are already in VIEW mode, issuing the VIEW command changes the display on the screen.
VSPC	Begins a terminal session.

GLOSSARY

This glossary defines data processing terms used in this manual—including terms used only in VSPC as well as standard data processing terms. Terms not used in this manual are omitted.

American National Standard definitions are identified with asterisks (*).

IBM is grateful to the American National Standards Institute (ANSI) for permission to reprint its definitions from the *American National Standard Vocabulary for Information Processing* (Copyright © 1970 by American National Standards Institute, Incorporated), which was prepared by Subcommittee X3K5 on Terminology and Glossary of the American National Standards Committee X3.

access control: The alteration of normal VSPC file access conventions by use of the VSPC PROTECT and SHARE commands, and by the specification of passwords.

accounting unit: An installation-defined measure of VSPC system usage that is used for user accounting.

administrative command: A special command used by the VSPC administrator to process user profiles and perform other supervisory tasks.

AID: *See* VSPC AID.

alphameric character: The letters A through Z, digits 0 through 9, punctuation marks, and #, $, and @.

alphanumeric: *See* alphameric character.

APL: A programming language. A problem-solving programming language designed for use at terminals. It has capabilities for handling arrays and for performing mathematical functions. VS APL is an IBM program product that can be used with VSPC.

attention key: A key on some terminals that interrupts execution by the Central Processing Unit.

attribute: A descriptor of the kind of information contained in a user's workspace of files. A user profile is assigned a default attribute that applies to the user's workspace when he logs on to VSPC.

background: The portion of VSPC that provides batch processing capabilities.

BASIC: An algebra-like programming language used for problem solving by engineers, scientists, and others who may not be professional programmers. VS BASIC is an IBM program product that can be used with VSPC.

batch job: A serially processed job, as opposed to an interactive job.

batch processing: The processing of computer jobs serially, as opposed to interactive processing.

blank: A nonprinting graphic character used to separate words or other meaningful information.

buffer: An area of storage that is temporarily reserved for use in performing an input/output operation, into which data is read or from which data is written. Synonymous with I/O area.

bulk input area: In VIEW mode, the area of the screen in which several workspace lines can be entered before interacting with the VSPC system to have the lines placed in the user's workspace.

byte: (1)*A sequence of adjacent binary digits operated upon as a unit and usually shorter than a computer word. (2) The representation of a character.

carrier return: The operation that prepares for the next character to be printed or displayed at the first position on the next line.

central processing unit (CPU): The unit of a computer that includes the circuits controlling the interpretation and execution of instructions.

chaining: A method of defining VSPC command lists in such a way that execution of one command list automatically begins execution of another.

character: *A letter, digit, or other symbol that is used as part of the organization, control, or representation of data.

character set: The set of unique characters associated with a coding system.

character string: *A string consisting solely of characters.

command: An imperative statement used interactively, at a terminal, to tell VSPC to perform a specific action.

command list: A sequence of VSPC commands that can be saved for future execution as a group.

command mode: A VSPC processing mode that allows the user to enter general VSPC commands in the input area.

communicating (online): Pertaining to a terminal connected by a wire or a telephone circuit with the central processing unit.

compilation: The process by which a language processor source program is translated into a machine language object program.

compile: To prepare a machine language program from a computer program written in another programming language.

compile time: The time during which a language processor source program is translated into a machine language object program.

compiler: *A program that compiles.

computer: *See* computing system.

computing system: A central processing unit with main storage, input/output channels, control units, storage devices, and input/output devices connected to it.

CONTINUE file: A special VSPC file, available to a VSPC user, in which the contents of the workspace are automatically placed when a forced ending occurs or when the user requests it during logoff.

conversational: Describing a program or a system that carries on a dialog with a remote terminal user, alternately accepting input and then responding to the input quickly enough for the user to maintain his train of thought.

current line pointer: A line number within a workspace that is maintained by VSPC, that is set by commands that change the workspace, and that may be symbolically referenced in VSPC commands.

cursor: A movable spot of light on the cathode ray tube of a console or display unit that indicates where the next character will be entered.

cursor control keys: Special keys on the 3270 used to move the cursor vertically and horizontally.

cursor wrap: Movement of the cursor off one edge of the screen, "around the back," and appearing on the opposite edge.

data: *A representation of facts, concepts, or instructions in a formalized manner suitable for communication, interpretation, or processing by humans or automatic means.

default: The choice among exclusive alternatives made by VSPC when no explicit choice is specified by the user.

dial line: *See* switched line.

dial-up terminal: A terminal on a switched network.

direct file: In VSPC a file containing records of fixed length with line numbers beginning at one and incrementing by one. In such a file, specific records can be retrieved using the line number as a search key, but not by VS BASIC. A direct file must have the DATA attribute.

directory: A set of records in the VSPC library, each one of which identifies and describes a file in a user library. Each user library has its own directory.

display station: The 3270 group of screens, printers, keyboards, and auxiliary equipment.

editable data: One or more lines, each consisting of a line number followed by a character string; each line can be acted upon by the VSPC data editing commands. Editable data can consist of processor source programs, command lists, or simple data without any processing attribute.

EOB: End of block; a code that marks the end of a block of data.

EOM: End of message; the specific character or sequence of characters that indicates the termination of a message or record.

external VSAM file: A VSAM file stored outside of the VSPC library that is available to VSPC users through language processor input/output statements.

file: A unit of information stored in a library (a library member). it is the unit of information stored in a user library.

file owner: The VSPC user who created a file. In a private library, the file owner and the library manager are the same user; in project and public libraries, they may be either the same or different users.

filename: A name that identifies a file. A filename consists of a number (libnum), a name, and possibly a password.

folding: A technique used with the universal character set feature to allow each of the 256 possible character codes to print some character on a chain or train with few graphics. For example, it allows the printing of uppercase graphics when lowercase graphics are not available in the character array.

forced ending: A terminal session ending that is not under the control of the terminal user. A forced ending can occur when there is a detectable communication breakdown between the terminal and the computer system, when the system operator halts VSPC, or when the VSPC operator halts the user's session.

foreground: The portion of VSPC that provides interactive processing capabilities.

foreground processor: A computer program, such as a language processor, that runs interactively under the control of VSPC.

FORTRAN: FORmula TRANslating system. A programming language used primarily to write computer programs in arithmetic formulas. VSPC FORTRAN is an IBM program product that can be used with VSPC.

general command area: In VIEW mode, an area of the screen within which VSPC general commands may be entered with the ENTER key.

hardcopy: A printed representation of data.

I/O: *See* input/output.

incr: *See* increment.

increment: An increase in quantity or value.

input/output: A general term for the equipment used to communicate with a computer; also the data involved in such communication; commonly called I/O.

installation: A general term for a particular computing system, in the context of the overall function it serves and the individuals who manage it, operate it, apply it to problems, maintain it, and use the results it produces.

integer: An unsigned natural number.

interactive processing: Computer processing in which each entry from a terminal elicits a response from the computer.

interpreter: A language processor that translates and executes each source program statement before it translates and executes the next one. For example, VS APL.

interpreter workspace: A workspace that contains information produced by working under the control of an interpreter. Also a file that contains such information.

invalid output: Characters not recognized by VSPC.

JCL: job control language

job card: A job control language statement that begins and identifies a job within the job stream.

job control language (JCL): A programming language used to code job control statements. Abbreviated JCL.

job control statement: *A statement in a job that is used in identifying the job or describing its requirements to the operating system.

job entry: The submission of a batch computer job from a terminal.

job stream: On input, the sequence of job control statements and data submitted to the batch operating system; on output, the diagnostic messages and other output data produced when a batch job is executed.

jobname: An eight-character name that identifies a VSPC user's batch job. The first six characters are the user's preassigned VSPC job entry code; the last two characters are numeric and assigned by VSPC.

keystroke: A pressing of a terminal key.

keyword: A VSPC command name or operand that identifies a specific action to be taken.

leased line: A direct connection between a terminal and a computer.

library: A collection of files. (*See* private library, project library, public library.)

library manager: The VSPC user to whom a user library was assigned at enrollment.

library member: A unit of information stored in a library (a file).

library number (libnum): The user number that identifies a user's private, project, or public library.

light pen: A light-sensitive pen that can detect characters displayed on a 3270 cathode ray tube. By pressing the tip of the pen on control line fields, the user can request specific VSPC actions.

line number: A one- through five-digit number that begins a line of editable data.

listing: A display or printout of data.

local command: In VIEW mode, a command that is entered over a line number in the workspace display area and that acts upon a limited number of lines.

logoff: A user's procedure for terminating a session with VSPC.

logon: A user's procedure for beginning a session with VSPC.

message: A combination of characters and symbols transmitted from one point to another on a telecommunication network.

message header: A six-character identifier for a VSPC message in the form ASUnnn (VS BASIC, ICDnnn), where nnn is a three-digit number. VSPC message headers are not normally printed; the user can, however, request that they be printed with each message displayed.

MTA (Multiple Terminal Access): A feature of some terminals allowing more than one terminal on one communication line. The NCP recognizes terminals by analysis of characters transmitted from the terminal.

NCP: *See* network control program.

network control program (NCP): A program, transmitted to and stored in a communications controller, that controls the operation of the communications controller.

nondial line: *See* leased line.

nonswitched line: *See* leased line.

null line: A VSPC line that contains no characters. A null line is entered by pressing carrier return before any characters have been entered or after all characters have been erased.

object program: An executable computer program resulting from the compilation of a source program.

offline: Pertaining to a terminal not communicating with the CPU.

online: Pertaining to a terminal communicating with the CPU.

operand: Information entered after a command name that modifies the action of the command and/or defines the data on which the command operates. (*See* keyword, variable operand.)

operating system: *Software which controls the execution of computer programs and which may provide scheduling, debugging, input/output control, accounting, compilation, storage assignment, data management, and related services.

operator: A member of a data processing installation who is responsible for directing the overall operation of a computing system.

option: An operand you need not state and is so designated.

parameter: A variable that is given a constant value for a specific purpose or process.

password: A string of one through eight characters that limits access to VSPC (logon password) or to a VSPC file (filename password).

PF key: See program function key.

predefined path: A direct connection from a terminal to the CPU, making special identification from the 2741 and 1050 at logon unnecessary.

private library: A VSPC library containing files that are normally available only to the user with that library number (user number); if this user so specifies, however, such files may be made available on a read-only basis to other VSPC users.

processing unit: * A unit of a computer that includes circuits controlling the interpretation and execution of instructions. Synonymous with central processor, main frame.

processor: A computer program that runs either interactively under the control of VSPC (*foreground processor*), or as a separate batch job that communicates with VSPC (*auxiliary processor*).

profile: *See* user profile.

program function key: A key on the keyboard of a display device that passes a signal to a program to call for a particular program operation.

program product: A licensed IBM program that performs a function for the user and usually interacts with and relies upon system control programming or some other IBM-provided control program. For example, VS APL, VS BASIC, and VSPC FORTRAN are IBM program products that rely on VSPC.

programming language: A language used to write computer programs, such as VS APL, VS BASIC, or VSPC FORTRAN.

project library: A VSPC library containing files that are normally available only to a subset of VSPC users whose profiles specify that they may have access to the library on a read-only basis. Files in this library may be modified only by their owners. Files may be removed only by the owner or the library manager. Ownership of files may be transferred among project users.

prompt: A request from VSPC for certain information, such as a password.

public library: A VSPC library containing files that are normally available to all VSPC users on a read-only basis. Files in this library may be modified only by their owners. Files may be removed only by the owner or the library manager.

remote terminal: *See* terminal.

S/S mode: start/stop mode

SDLC mode: synchronous data link control mode

selector light pen: *See* light pen.

sequential file: A file whose records are organized on the basis of their successive physical positions, such as on magnetic tape. *Contrast with* direct file.

session: The activity that occurs at a terminal between logon and logoff.

shared storage: Virtual storage used by VSPC for data exchange among users and auxiliary processors.

source program: A sequence of computer program statements written in a programming language and requiring compilation or interpretation to be executable by the computer.

special character: *A graphic character that is neither a letter, nor a digit, nor a space character.

start/stop mode (S/S mode): Asynchronous transmission between a terminal and the computer in which each character (or group of code) is preceded by a start signal and followed by a stop signal. Abbreviated S/S mode.

string: *A linear sequence of entities such as characters or physical elements.

stylus: *See* light pen.

switched line: A communication line in which the connection between the computer and a remote station is established by dialing. Synonymous with dial line.

synchronous data link control mode (SDLC mode): Synchronous transmission between a terminal and the computer in which the sending and receiving instruments are operating continuously. Abbreviated SDLC mode.

syntax: *The structure of expressions in a language.

terminal: A device, usually equipped with a keyboard and some kind of display, capable of sending and receiving information over a communication channel.

terminal user: *See* user.

time sharing: A method of using a computing system that allows a number of users to execute programs concurrently and to interact with the programs during execution.

typamatic: Pertaining to typewriter keys having the ability to repeat their character or function when held down.

type element: A hardware feature on some terminals (2741, 1050) that is shaped like a ball and moves horizontally across the terminal on a carrier, printing characters on the paper as the user presses the keys.

undefined file: A VSPC processor output file that has records in a format unrecognizable by VSPC; such files cannot be edited using VSPC.

user: Under VSPC, anyone eligible to log on.

user identification (usernum): *See* user number.

user number: A one- to seven-digit symbol identifying each system user.

user profile: The definition of the characteristics associated with a specific end user, including: the user number, the optional password, whether or not this user is a project or public library manager, maximum storage allocation for this user's library and workspace, the default workspace contents, whether or not this user is able to use job entry commands, and whether or not this user is privileged to use administrative commands.

usernum: *See* user number.

variable operand: An operand that modifies the action of a VSPC command through a user-chosen value.

VIEW mode: A VSPC display terminal processing mode in which the user can modify a displayed workspace using the local facilities of the terminal, and using local and general commands, and can then copy changes back into the workspace.

visual fidelity: A technique that uses the position of the print head as a guide to where on the input line the next action will take place.

VS APL: An APL language running under VSPC.

VS BASIC: *See* BASIC.

VS Personal Computing: A program product that allows interactive or batch processing from a terminal (abbreviated VSPC).

VSAM: Virtual storage access method. The set of programs the operating system uses to transfer data between VSPC and direct-access storage.

VSPC: VS (Virtual System) Personal Computing

VSPC administrator: A VSPC user authorized to use the administrative commands for enrolling new VSPC users and processing their user profiles.

VSPC AID: A program contained in VSPC that, on request, prompts a user to specify a VSPC command correctly or provides an explanation of a VSPC message.

VSPC FORTRAN: A FORTRAN language running under VSPC.

VTAM: Virtual telecommunications access method. The set of programs the operating system uses to transfer data between terminals and VSPC.

workspace: The storage in which a VSPC user does his terminal work (such as entering data or source program statements and compiling and executing programs).

workspace attribute: The content classification of a portion of storage in VSPC.

workspace display area: An area of a display screen in VIEW mode in which part of the user's workspace is displayed and can be changed by use of the local facilities of the terminal.

wrap: *See* cursor wrap.

INDEX

This is a combined index containing entries for this publication and for *VS BASIC Language,* GC28-8303. Entries with page numbers refer to pages in this publication; entries followed by an asterisk (*) refer to information found in the language publication.

Symbols

A

B

C

D

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Numeric

Reader's Comment Form

VS BASIC for VSPC: Terminal User's Guide
SH20-9060-2

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate. Comments may be written in your own language; use of English is not required.

IBM shall have the nonexclusive right, in its discretion, to use and distribute all submitted information, in any form, for any and all purposes, without obligation of any kind to the submitter. Your interest is appreciated.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Note: Staples can cause problems with automated mail sorting equipment. Please use pressure sensitive or other gummed tape to seal this form.

Fold on two lines, tape, and mail. No postage necessary if mailed in the U.S.A. (Elsewhere, any IBM representative will be happy to forward your comments.) Thank you for your cooperation.

SH20-9060-2

Reader's Comment Form

Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

IBM Corporation
P.O. Box 50020
Programming Publishing
San Jose, California 95150

Fold and Tape

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601

VS BASIC for VSPC: Terminal User's Guide Printed in U.S.A. SH20-9060-2

SH20-9060-2

VS BASIC for VSPC: Terminal User's Guide Printed in U.S.A. SH20-9060-2

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601