Apple III

COBOL
Quick Reference Guide

Apple III COBOL

Quick Reference Guide

Acknowledgements

COBOL is an industry language and is not the property of any company or group of companies, or of any organization or group of organizations.

No warranty, expressed or implied, is made by any contributor or by the CODASYL Programming Language Committee as to the accuracy and functioning of the programming system and language. Moreover, no responsibility is assumed by any contributor, or by the committee, in connection herewith.

The authors and copyright holders of the copyrighted material used herein:

> FLOW-MATIC (Trademark for Sperry Rand Corporation) Programming for the Univac® I and II, Data Automation Systems copyrighted 1958, 1959, by Sperry Rand Corporation; IBM Commercial Translator Form No. F28-8013, copyrighted 1959 by IBM; FACT, DSI27A5260-2760, copyrighted 1960 by Minneapolis-Honeywell.

have specifically authorized the use of this material in whole or in part, in the COBOL specifications. Such authorization extends to the reproduction and use of COBOL specifications in programming manuals or similar publications.

Contents

Introduction

This Quick Reference Guide is a compact summary of the information required to develop software using the Apple III COBOL System. Comprehensive descriptions of the language and operating system features are contained in the following documents:

Apple III COBOL Introduction and Operating System Manual

Apple III COBOL Language Reference Manual

Run-Time Commands

SOS Control Keys

- CONTROL-C (ASCII ETX) signals end of file for console input.
- CONTROL-X (ASCII CAN) erases all of the current line.
- CONTROL-\ aborts a program and returns control to the main COBOL command line. Requires a following RETURN if the system is processing an ACCEPT statement.
- Numeric Keypad Controls:
- CONTROL-5 toggles CRT screen refresh; any output to the console between CONTROL-5's will be lost.
- CONTROL-6 erases any characters typed ahead (and not yet processed as input by a program).
- CONTROL-7 toggles console acceptance of output; the program halts temporarily until the next CONTROL-7 allows output to resume where it was stopped.
- CONTROL-8 toggles the visible representation on the CRT of control characters in console output.

COBOL Command Line Summary

A	A(nimate. Type the pathname of the program to be animated (extension .INT assumed if none given). Type ! to exit.
C	C(ompile. Type the pathname of the source code file (default extension .CBL) and Compiler directives. Type ! to exit.
F	F(orms2. Invokes the Apple III FORMS2 utility. Type ! to exit.
Q	Q(uit. Exits COBOL.
R	R(un. Type the pathname of the intermediate code file (default extension .INT; terminate the pathname with an extra period for a file without .INT extension).
S	S(witches. Displays current settings of Run-Time switches: "-" if clear, digit or letter "A" if set. Type C to clear all switches, or type the digit or letter to toggle the current setting.
U	U(tilities. COBOL utilities:

C(opy — makes a duplicate of a file. Copy's name can differ from the original.

D(ate — sets Apple III date and time. Displays current date and time. Type over any field to change its value.

L(ist-dir — lists an Apple III disk directory.

E(xt-dir — lists a disk directory and subdirectories.

P(refix — sets Apple III prefix.

R(emove — removes a file. Requires Y to confirm removal.

T(ype — lists a file on the console.

Q(uit — exits to the main COBOL command line.

Backing over characters with the LEFT-ARROW key doesn't erase them; pressing RETURN sends the entire visible line to the utility.

Compiler Directives

The general form of the command line for a compilation is:

```
file name [ directive ... ]
```

where the possible directives are listed below. Default settings are indicated with an asterisk.

*ANIM	Specifies output of files for use of Animator.
NOANIM	Turns off specification of Animator file output.
BRIEF	Specifies omission of text from error messages.
*NOBRIEF	Explanatory text listed with each error message.
COMP	Specifies PIC 99 and PIC 9(4) COMPUTATIONAL data are binary; MOVE and ON SIZE ERROR treated in non-standard manner.
*NOCOMP	PIC 99 and PIC 9(4) items as in standard COBOL.
COPYLIST	Specifies that the source code of COPY files is to be listed.
*NOCOPYLIST	COPY file source code is not listed.
*CRTWIDTH	(default=128) Logical line size for ANSI ACCEPT and DISPLAY.
NOCRTWIDTH	Specifies no ANSI ACCEPT/DISPLAY; frees up table space.
*DATE	Followed by a character string in parentheses, replaces the entire comment-entry in the program's DATE-COMPILED paragraph.
NODATE	Suppresses replacement of comment-entry in DATE-COMPILED paragraph.
*ECHO	Writes error messages to the display.
NOECHO	Suppresses display of error messages.
ERRLIST	Writes to the listing file only lines with syntax errors.
*NOERRLIST	Generates a full listing of the source code.

FLAG	Specifies output of validation flags at compile time: LOW, L-I, H-I, HIGH, A///, IBM
*NOFLAG	No validation flags are listed at compile time.
*FORM	(default=60) Sets number of lines per page in the list file.
NOFORM	Specifies that no form or page headings are to be generated.
IBM	Enables compilation of certain IBM extensions to ANSI COBOL.
*NOIBM	Causes the compiler to treat these as errors.
*INT	Specifies the name of the intermediate code file output (default is basename of source file with extension .INT).
NOINT	Specifies that no intermediate code file is to be generated.
*LIST	Specifies the name of the listing file output, or if no file name is given, produces a listing on the console display (default is disk file listing with extension .LST).
NOLIST	Specifies that no listing file is to be generated.
*PRINT	Synonymous with LIST.
NOPRINT	Synonymous with NOLIST.
REF	Specifies the listing of 4-digit hexadecimal addresses.
*NOREF	Suppresses the output of location addresses.
RESEQ	Specifies resequencing of source lines in increments of 10.
*NORESEQ	No alteration of columns 1 through 6 on listing.
FORMFEED SYSIN SYSOUT TAB	Enables use of its character string parameter as the name associated with this function in the SPECIAL-NAMES paragraph.

Animator Command Summary

A lOcAte command. Finds the declaration of a data-name or procedure-name specified by typing in the name; see also the "O" command.

B Breakpoint command. Sets COBOL statement at which execution will halt:

- S Set breakpoint at statement currently pointed to by the cursor.
- U Unset (clear) the breakpoint currently pointed to by the cursor.
- C Cancel all breakpoints.
- X eXamine next breakpoint. Use successively to find all breakpoints.

C Compile command. Compiles and executes COBOL statements typed in during the debugging session.

D Display command. Displays and optionally modifies named data-item.

E Execute command. Specifies execution option:

- X eXecute one COBOL statement; move cursor to the next statement.
- K sKip one COBOL statement; move cursor to the next statement.
- I executes to the next If statement; halts and positions the cursor at this statement.
- G (Go) start continuous Animation; speed of animation set by typing a numeric character from 1 (slowest) to 9 (fastest).
- Z (Zoom) start execution without Animation (normal execution).
- S Stop execution; display the current user screen.

F Find command. Searches from the current cursor position through the source text for a specified string of characters.

L Level command. Sets "threshold" level for nested PERFORM statements; PERFORMs subordinate to this level are treated as a single statement.

M	Monitor command. Starts automatic display of a data-item after each statement executed during Animation.

N Name command. Specifies which programs are executed with Animation:

W	Which program: displays the current program name.
A	All programs: (default) all programs compiled with the ANIM directive will be run under Animation.
T	This program: only the current program will be Animated; all others will execute normally.
O	Other program: type the name of another program. The current program compeletes execution under Animation; then until the named program is called, execution proceeds normally. Animation resumes at the start of the named program.

O lOcAte command. Finds the declaration of the data-name or procedure-name that the cursor is resting on when "O" is typed. See also the "A" command.

P Program-counter command. Displays or changes the point of execution:

W	Where : repositions the display window to show the statement at the hexadecimal address specified.
R	Resets the execution start point to the current cursor position.

Q Query command. Displays or changes the value of the data-item on which the cursor is resting (may not be used for condition-names).

S Screen command. Repositions the screen window as follows:

N	displays Next screen from source text.
P	displays Previous screen from source text.
T	displays screen at Top of source text.
E	displays screen at End of source text.
V	repositions window so that the source line indicated by the cursor is on the third line.

H	splits the screen in Half (i.e., into two windows) with a dividing line of hyphens. The lower window is positioned to show the top of source text. Note: Subsequent screen commands, operate in the window in which the cursor is positioned.
F	restores Full screen display (single window).
〉	(also unshifted "."") displays next screen from source text.
〈	(also unshifted ",") displays previous screen from source. These are like N and P but available at outermost level.
=n	repositions the window such that the nth source line is aligned at the third screen line.
+n	moves the window forward n lines.
−n	moves the window back n lines.

Note: =, +, − all position the cursor for entry of a numeric quantity followed by RETURN.

When S is typed with the cursor resting on the dividing line in a split screen display, the options available are

U	moves the screen divider Up one line.
D	moves the screen divider Down one line.

T unTil command. Sets condition for execution to halt:

S	Set. Type a COBOL conditional to be tested after each statement; execution halts when the test passes.
U	Unset (clear) the previously set condition.
X	Display the previously set conditional expression.

U User command. Displays the current user screen, replacing the source code window display until any key is pressed.

Z Zoom command. Specifies continuation of execution of the program without further invocation of the Animation Option.

FORMS2 Command Summary

Initialization

1. DATA-NAME & FILE-NAME. Mandatory one to six character name to be used throughout the run as the base of all file names and COBOL data-names. "!" instead of name exits from FORMS2.
2. CRT lines. Default value 24; optionally 22 or 23.
3. CURRENCY SIGN. Default value "$".
4. DECIMAL-POINT. Default value ".".
5. Output files generated. The options are:

A	DDS file of COBOL Data-Description Statements only.
B	DDS file and CHK program to checkout the forms.
C	DDS and CHK files, and Snn screen image files.
D	DDS and Snn files only.
E	Snn files only.
F	No files output.
G	DDS and Snn files, and GEN index-file program.

6. DEVICE/DIRECTORY PREFIX. Default is none. Zero to forty characters, used as a prefix to the name base. DDS for COPY statements.

Work Phase Initialization

From screen W01, choose one of the following options, or request help (press the "?" key) or exit from FORMS2 (by pressing "!").

A Fixed Text on Clear Screen. Input defines a new COBOL record comprised of FILLER entries in blank spaces and PIC X(n) data with VALUE clauses defined by edit mode data.

B Fixed Text on Last Screen. Like A but the new record REDEFINES the previous screen record.

C	Variable Data on Last Screen. Input "X"s, "Y"s, "8"s, "9"s and Numeric-editing characters against the background of the previous screen. Output is a record which REDEFINES the last one and can be used in ACCEPT statements protecting the rest of the screen.
D	Variable Data without Redefinition. Like C, but the record is not a redefinition of the background.

General Commands

!	(Exclamation point) Exit from FORMS2 run. Normally issued at the reappearance of the W01 work-phase initialization screen, to indicate that no more forms are to be generated.
?n ?	(n = 1, 2, 3 or 4) Display HELP screen number n; if n is omitted, display H01 or the next help screen in sequence.
_	(Underline) Resume edit mode. This is the default command.
*	(Asterisk) Mark the boundary between key field and data in the Variable Data record generated for an Index File program.
b̸	(Blank) Release the current form for processing, to end this work phase. Not accepted when generating index file program, unless "*" command marks the key/data boundary.
An	(n = 1 to 9) Duplicate the current line n times, below the cursor; the cursor must be at the start of the line.
Cn	(n = 1 to 9) Insert n blank characters at the cursor position.
Dn	(n = 1 to 9) Delete n characters starting at the cursor.
F	Display the Foreground/Background menu (work screen W02). Allows selection of commands FA through FJ by pressing keys A through J, returning to the W02 menu after each command, until option A (return to edit-mode) is selected. All the options may be selected without reference to the menu by the following two-keystroke commands:

FA	Return from Foreground/Background manipulation to edit-mode.
FB	Clear Foreground; erases current Foreground screen, leaving the Background unaffected.
FC	Clear Background; erases current Background screen, leaving the Foreground and any records already generated unaffected.
FD	Overlay Background data onto Foreground; places the entire Background screen into the Foreground.
FE	Overlay Foreground data onto Background; places Foreground screen contents into the Background.
FF	Overlay Screen Image File onto Foreground. Requests name of the file, then reads it into the Foreground screen.
FG	Overlay Screen Image File onto Background. Requests name of the file, then reads it into the Background.
FH	Display Foreground; shows Foreground contents, without merging the Background contents into it.
FI	Display Background; shows Background contents not merged with Foreground.
FJ	Display Screen Image File; requests the name of the file and displays it on the screen.
G or G0	Generate screen coordinates names; changes the names in the COBOL record outputs, so that data-items have row and column position appended instead of sequential field number.
G1	("gee one") Restores default naming.
In	(n = 1 to 9) Insert n blank lines before the current line; moves the current line and subsequent lines down the screen.

Jn	(n = 0 to 9) Multiple space reset; initial setting 1. All blank areas on a fixed screen between visible characters are replaced by FILLER items, whenever the number of contiguous spaces is greater than n; if n=0, all spaces become FILLER.
Kn	(n = 1 to 9) Kill (delete) n lines, starting with the current line; if there are fewer than n lines, all remaining lines are deleted.
Mx	(x any printable character; initial setting "_") Make "x" the "visible-space" character: use of this character in edit-mode causes creation of a blank in the COBOL value clause describing the item.
O	(letter "oh") Turn on automatic screen preparation.
O1	("oh one") Turn off automatic screen preparation.
P	Show current cursor position (yyxx for row yy, column xx).
Q	Quit: during initialization, returns to the first screen for revision of the parameters already selected; during a work phase, returns to the W01 work initialization, for revision of the type of record being generated.
S0	Cancel S option (S3 or S9) in effect.
S1	Suppress COBOL statement generation for current work phase.
S2	Suppress screen image generation for current work phase.
S3	Request user names for all screen image files.
S9	Edit pause for each COBOL line output; allows minor editing within lines of the COBOL DDS records.
Un	(n = 1 to 9) Move the cursor vertically upwards n lines, leaving it in the same column as on the starting line.
Vn	(n = 1 to 9) Move the cursor vertically downward n lines, leaving it in the same column as on the starting line.

W or WO	Window "home" key; positions the cursor at the start of the top line in the current window. "W" is synonymous with "WO".
W1	Define starting line of a window at the current line; shows a line of delimiters ("-"s) on the previous line.
W2	Define end line of window as the current line; shows a line of delimiters on the next line.
W3	Define starting line of window; no delimiting line shown.
W4	Define end line of window; no delimiting line shown.
W5	Display delimiters on the line before the current window.
W6	Display delimiters on the line after the current window.
W7	Erase any delimiter line before the current window; restores any work screen contents previously obscured by delimiters.
W8	Erase any delimiter line after the current window.
W9	Position cursor at the end of the current window.
X	Reposition the command line to the current cursor position.

Compiler Error Message Summary

Error	Description
01	Compiler error; consult Technical Support
02	Illegal format: Data-name
03	Illegal format: Literal, or invalid use of ALL
04	Illegal format: Character
05	Data-name not unique
06	Too many data or procedure names declared, or insufficient memory
07	Illegal character in column 7, or continuation error
08	Nested COPY statement, or unknown COPY file specified
09	"." missing
10	Statement starts in wrong area of source line
21	"." missing
22	DIVISION missing
23	SECTION missing
24	IDENTIFICATION missing
25	PROGRAM-ID missing
26	AUTHOR missing
27	INSTALLATION missing
28	DATE-WRITTEN missing
29	SECURITY missing
30	ENVIRONMENT missing
31	CONFIGURATION missing
32	SOURCE-COMPUTER missing
33	OBJECT-COMPUTER/SPECIAL-NAMES clause error
34	OBJECT-COMPUTER missing
36	SPECIAL-NAMES missing
37	SWITCH clause error, or system name/mnemonic name error
38	DECIMAL-POINT clause error
39	CONSOLE clause error
40	Illegal currency symbol
41	"." missing
42	DIVISION missing
43	SECTION missing

44	INPUT-OUTPUT missing
45	FILE-CONTROL missing
46	ASSIGN missing
47	SEQUENTIAL or RELATIVE or INDEXED missing
48	ACCESS missing on indexed/relative file
49	SEQUENTIAL or DYNAMIC missing or 〉64 alternate keys
50	Illegal ORGANIZATION/ACCESS/KEY combination
51	Unrecognized phrase in SELECT clause
52	RERUN clause syntax error
53	SAME AREA clause syntax error
54	Missing or illegal file-name
55	DATA DIVISION missing
56	PROCEDURE DIVISION missing or unknown statement
57	Program collating sequence not defined
61	"." missing
62	DIVISION missing
63	SECTION missing
64	File-name not specified in SELECT statement, or invalid CD name
65	RECORD SIZE integer missing, or line sequential record 〉1024 bytes
66	Illegal level no. (01-49), 01 level required, or level hierarchy wrong
67	FD, CD or SD qualification syntax error
68	WORKING-STORAGE missing
69	PROCEDURE DIVISION missing, or unknown statement
70	Data description qualifier or "." missing
71	Incompatible PICTURE clause and qualifiers
72	BLANK illegal with non-numeric data-item
73	PICTURE clause too long
74	VALUE with non-elementary item, wrong data-type or value truncated
75	VALUE in error or illegal for PICTURE type
76	Non-elementary item has FILLER/SYNC/JUST/BLANK clause
77	Preceding item at this level has 〉8192 bytes or 0 bytes
78	REDEFINES of unequal fields or different levels
79	Data storage exceeds 64K bytes

81	Data description qualifier inappropriate or repeated
82	REDEFINES data-name not declared
83	USAGE must be COMP, DISPLAY or INDEX
84	SIGN must be LEADING or TRAILING
85	SYNCHRONIZED must be LEFT or RIGHT
86	JUSTIFIED must be RIGHT
87	BLANK must be ZERO
88	OCCURS must be numeric, non-zero, unsigned or DEPENDING
89	VALUE must be literal, numeric literal or figurative constant
90	PICTURE string has illegal precedence or illegal character
91	INDEXED data-name missing or already declared
92	Numeric-edited PICTURE string is too large
101	Verb not recognized or "." missing
102	IF....ELSE mismatch
103	Operand missing or has wrong type or undeclared, or "." missing
104	Procedure name not unique, or USE procedure duplicated
105	Procedure name same as data-name
106	Name required
107	Wrong combination of data-types
108	Conditional statement not allowed in this context
109	Malformed subscript
110	ACCEPT/DISPLAY wrong or Communications syntax incorrect
111	Illegal syntax used with I-O verb
112	Invalid arithmetic statement
113	Invalid arithmetic expression
115	Invalid conditional expression
116	IF statements nested too deep, or too many AFTERs in PERFORM statement
117	Incorrect structure of PROCEDURE DIVISION
118	Reserved word missing or incorrectly used
119	Too many subscripts in one statement
120	Too many operands in one statement
141	Inter-segment procedure name duplication
142	IF....ELSE mismatch at end of source input
143	Operand has wrong data-type or not declared
144	Procedure name undeclared

145	INDEX data-name declared twice
146	Bad cursor control: illegal AT clause
147	KEY declaration missing or illegal
148	STATUS declaration missing
149	Bad STATUS record
150	Undefined inter-segment reference, or error in ALTERed paragraph
151	PROCEDURE DIVISION in error
152	USING parameter not declared in LINKAGE SECTION
153	USING parameter not level 01 or 77
154	USING parameter used twice in parameter list
155	FD missing
157	Incorrect structure of PROCEDURE DIVISION
160	Too many operands in one statement

In addition to these numbered error messages, the following message can be displayed with subsequent termination of the compilation:

FATAL I-O ERROR: file name

where file name is the erroneous file. Any intermediate code file produced in such a case is not usable. The conditions that will cause this error are

Disk overflow
File directory overflow
File full
Impossible I-O device usage

Other operating system dependent conditions may also cause this error.

Run-Time System Error Message Summary

General Run-Time Errors

150	Program interrupted by user
153	Subscript bounds overflow: zero or greater than the number of occurrences of the item
154	PERFORMs nested too deep: usually results from using GO TO to jump out of the range of a PERFORM instead of jumping to an EXIT statement at the end of its range
157	Not enough program memory: may occur on initial program load or when the Run-Time System attempts to load one of its own modules to perform a function such as indexed I-O, SORT/MERGE or ACCEPT/DISPLAY on CRT—see the ON OVERFLOW clause of the CALL statement for handling sub-programs that can't be loaded
160	Overlay loading error: unable to load overlay or segment; for example, file not found, too many files open, or invalid file structure
161	Illegal intermediate code: operation not recognized by the Run-Time System—implies bad program file
162	Perform n times nested too deep: too many levels of PERFORM n TIMES. Error may be reported in processing a complex arithmetic expression
163	Program counter out of range: address in GO TO, PERFORM or ALTER lies outside the program area—implies bad program file
164	Program not found: loading error (for example, file not found, too many files open, invalid file structure)
165	Version number error: incompatible releases of Compiler and Run-Time System; the Compiler used may have generated code that will not be executed correctly
166	Recursive call illegal: attempt to CALL a COBOL module recursively (i.e., when it is already active)
167	Too many USING items: the list of items supplied in a CALL ... USING statement is longer than the Run-Time System can handle
168	Linkage Error: parameter count mismatch between CALL and PROCEDURE DIVISION USING statements, or an attempt to access a linkage section item when a program executes directly

	or when the item isn't included in the PROCEDURE DIVISION USING list
174	ISR file loading error: Intersegment Reference File for a segmented program cannot be loaded; for example if the file was not found, or had an invalid file structure
176	Illegal intersegment reference: illegal use of GO TO, PERFORM or ALTER across segment boundaries in a segmented program
177	Cancellation of active program. Attempt to CANCEL a COBOL module that is still active (it has been called but has not yet executed an EXIT PROGRAM statement)
178	Error during save: unable to SAVE the program successfully; for example, when not enough disk or directory space
200 to 255	Unclassified error condition: may be caused by a disk or directory structure error not checked for by the operating system—consult Technical Support if the problem is reproducible after transferring all files in use to another disk.

File Handling Errors

Error Number	Meaning	File Organization Applicable
1	Out of Buffer space. Insufficient memory available for operating system I-O buffers	All
4	Illegal file name. File or device name contains illegal character(s).	All
5	No such device. The device or disk specified cannot be found by the system	All
7	Out of disk space. No space available on disk for file creation/extension	All

9	Disk directory full. No space available in disk directory for further entries	All
13	File not found. The file specified cannot be found by the system (in attempting to open for input a non-existent file not declared OPTIONAL)	All
14	Too many files open. Attempt to open more files (16) than can be catered to by the system; note that segment changes in segmented programs and calls to non-resident subprograms require the Run-Time System to open a file to satisfy the request. May mean that the Run-Time System can't acquire the memory it needs for I-O buffers	All
15	Too many open ISAM files. Attempt to open more indexed files (8) than can be catered to by the system.	Indexed
16	Too many open devices. Attempt to open more devices than can be used simultaneously by the system	All
24	Hardware I-O error. Device or disk I-O error; for example, checksum error, read after write verification failure, parity error, etc.	All
25	Operating system data error. Bad directory entry, invalid block allocation map, etc.	All

37	File access denied. Access to file denied by operating system; for example, in an attempt to read from an output device, write to a write-protected file, etc.	All
38	Incompatible disk. Disk created under another operating system or operating system version, or clashes with one already loaded (same name, etc.)	All
39	Incompatible file. Directory entry indicates incorrect file type, device type illegal for file organization, etc.	All
41	Bad file. File corrupt or in unrecognized format. Possibly caused by opening a file with a different organization or record length from that used to create it. May occur if the file was not properly closed after a preceding update; for example, because of a hardware failure	Relative
42	Misformed line sequential file. A text file was opened and found to contain >0 and <1024 bytes. A normal text file has 1024 bytes of operating system data at the beginning.	Line Sequential
43	File information missing. Indexed files—means that one file is missing completely or that a file is shorter than indicated by its internal control data (generally caused by a failure to close the file after an update, for example because of a hardware failure)	Indexed

47	Index structure overflow. Indexed files—means that the maximum number of levels permitted in the index tree structure has been exceeded: the file must be reorganized before further data is added	Indexed
129	Record zero illegal. An attempt has been made to access record zero on a relative file	*Relative
139	Record length or key data error. Attempt to open an existing file where record length or key data differs from that used when it was created	**Line Sequential **Relative Indexed
141	File already open. Attempt to open a file that is already open	All
142	File not open. Attempt to close an unopened file	All
143	Rewrite/delete not preceded by read. Rewrite or delete on a file in sequential access mode was not preceded by a successful read	Sequential Relative Indexed
146	No current record. Sequential read attempted on a file in dynamic or sequential access mode when no current record was defined	Relative Indexed
147	Wrong open mode for read/start. Attempt to read from or start on a file that has not been opened input or I-O	All

148	Wrong open mode for write. Attempt to write to a file in sequential access mode that has not been opened output or extend, or attempt to write to a file in random or dynamic access mode that has not been opened input or I-O	All
149	Wrong open mode for rewrite/delete. Attempt to rewrite or delete on a file that has not been opened I-O	Sequential Relative

* — Means file is bad if reported for an indexed file.

** — Error may not be detected at open time but gives rise to a bad file when I-O is attempted

Table of Possible MOVEs in a COBOL Program

Category of Sending Data Item		Category of Receiving Data Item		
		Alphabetic	Alphanumeric, Alphanumeric Edited	Numeric Integer, Numeric Non-Integer, Numeric Edited
ALPHABETIC		Yes	Yes	No
ALPHANUMERIC		Yes	Yes	Yes
ALPHANUMERIC EDITED		Yes	Yes	No
NUMERIC	INTEGER	No	Yes	Yes
	NON-INTEGER	No	No	Yes
NUMERIC EDITED		No	Yes	Yes

SET Statement Valid Operations

SET TO FORMAT

		TO			
		Integer Literal	Integer data item	Index Name	Index Data Item
Identifier-1	Integer data item				X
	Index data item			X	X
Index-name-1		X	X	X	X

		Index data item	Elementary numeric integer	Greater than 0	Optional sign
SET TO Format	Identifier-1, -2	X	X		
	Integer-1			X	X
SET {UP / DOWN} BY Format	Identifier-3		X		
	Integer-2				X

Permissible I-O Statements and File OPEN Modes

ACCESS METHOD	STATEMENT	I-O MODULE									
		SEQUENTIAL OPEN MODE				RELATIVE OPEN MODE			INDEXED OPEN MODE		
		INPUT	OUTPUT	I-O*	EXTEND	INPUT	OUTPUT	I-O	INPUT	OUTPUT	I-O
Sequential	READ	X		X		X		X	X		X
	WRITE		X		X		X			X	
	REWRITE			X				X			X
	START					X		X	X		X
	DELETE							X			X
Random	READ					X		X	X		X
	WRITE						X	X		X	X
	REWRITE							X			X
	START								X		X
	DELETE							X			X
Dynamic	READ					X		X	X		X
	WRITE						X	X		X	X
	REWRITE							X		X	X
	START					X		X	X		X
	DELETE							X			X

* This OPEN mode not supported for ORGANIZATION LINE SEQUENTIAL

Reserved Word List

The following are reserved words in COBOL and Apple *///* COBOL.

The / symbol indicates that both the text up to that point and the whole word are reserved words. For example, in INDEX/ED, INDEX and INDEXED are reserved words.

ACCEPT
ACCESS
ADD
ADVANCING
AFTER
ALL
ALPHABETIC
ALSO
ALTER
ALTERNATE
AND
ARE
AREA/S
ASCENDING
ASSIGN
AT
AUTHOR

BEFORE
BLANK
BLOCK
BOTTOM
BY

CALL
CANCEL
CD
CHARACTER/S
CLOCK-UNITS
CLOSE
COBOL
CODE/-SET
COLLATING
COMMA
COMMUNICATION
COMP/UTATIONAL/-3
COMPUTE
CONFIGURATION
CONSOLE
CONTAINS
CONTAINS
COPY
CORR/ESPONDING
COUNT
CRT
CRT-UNDER
CURRENCY
CURSOR

DATA
DATE-WRITTEN
DATE/-COMPILED
DAY
DEBUG-CONTENTS
DEBUG-ITEM
DEBUG-LINE
DEBUG-NAME
DEBUG-SUB-1
DEBUG-SUB-2
DEBUG-SUB-2
DEBUGGING
DECIMAL-POINT
DECLARATIVES
DELETE
DELIMITED
DELIMITER
DEPENDING
DESCENDING
DESTINATION
DISABLE
DISPLAY
DIVIDE
DIVISION
DOWN
DUPLICATES
DYNAMIC

ELSE
ENABLE
END
ENTER
ENVIRONMENT
EQUAL
ERROR
EVERY
EXCEPTION
EXCESS-3
EXIT
EXTEND

FD
FILE
FILE-CONTROL
FILLER
FIRST
FOOTING
FOR
FORMFEED
FROM

GIVING
GO
GREATER

HIGH-VALUE/S

I-O/-CONTROL
IDENTIFICATION
IF
IN
INDEX/ED
INITIAL
INPUT/-OUTPUT
INSPECT
INSTALLATION
INTO
INVALID
IS

JUST/IFIED

KEY

LABEL
LEADING
LEFT
LESS
LIMIT/S
LINAGE/-COUNTER
LINE/S
LINKAGE
LOCK
LOW-VALUE/S

MEMORY
MERGE
MESSAGE
MODE
MODULES
MOVE
MULTIPLE
MULTIPLY

NATIVE
NEGATIVE
NEXT
NOT
NUMERIC

OBJECT-COMPUTER
OCCURS
OF
OFF
OMITTED
ON
OPEN

OPTIONAL
OR
ORGANIZATION
OUTPUT
OVERFLOW

PAGE
PERFORM
PIC/TURE
POINTER
POSITIVE
PROCEDURE/S
PROCEED
PROGRAM/-ID

QUEUE
QUOTE/S

RANDOM
RD
READ
RECEIVE
RECORD/S
REDEFINES
REEL
REFERENCES
RELATIVE
RELEASE
REMAINDER
REMOVAL
RENAMES
REPLACING
RERUN
RETURN
REWRITE
RIGHT
ROUNDED
RUN

SAME
SD
SEARCH
SECTION
SECURITY
SEGMENT/-LIMIT
SELECT
SEND
SENTENCE
SEPARATE
SEQUENCE
SEQUENTIAL
SET

SIGN
SIZE
SORT
SORT-MERGE
SOURCE/-COMPUTER
SPACE/S
SPECIAL-NAMES
STANDARD/-1
START
STATUS
STOP
STRING
SUB-QUEUE-1
SUB-QUEUE-2
SUB-QUEUE-3
SUBTRACT
SWITCH
SYMBOLIC
SYNC/HRONIZED
SYSIN
SYSOUT

TAB
TABLE
TALLYING
TAPE
TERMINAL
THAN
THEN
THROUGH
THRU
TIME/S
TO
TOP
TRAILING
TYPE

UNIT
UNSTRING
UNTIL
UP
UPON
USAGE
USE
USING

VALUE/S
VARYING

WHEN
WITH
WORDS
WORKING-STORAGE
WRITE

ZERO/ES or S

. (period)
(
-
*
**
)
;
+
/
,
<
=
>

Syntax Summary

All the syntax for Apple III COBOL is summarized below.

Shading denotes that the feature is an Apple III COBOL extension to ANSI COBOL.

D denotes that the feature serves only a documentary purpose in Apple III COBOL.

GENERAL FORMAT FOR IDENTIFICATION DIVISION

```
IDENTIFICATION DIVISION.

PROGRAM-ID.       program name

[AUTHOR.               [comment entry] ...]

[INSTALLATION.         [comment entry] ...]

[DATE-WRITTEN.         [comment entry] ...]

[DATE-COMPILED.        [comment entry] ...]

[SECURITY.             [comment entry] ...]
```

GENERAL FORMAT FOR ENVIRONMENT DIVISION

```
ENVIRONMENT DIVISION.

CONFIGURATION SECTION.

SOURCE-COMPUTER.  source-computer-entry  [WITH DEBUGGING MODE].

OBJECT-COMPUTER.  object-computer-entry

    [                           { WORDS      } ]
    [ ,MEMORY SIZE integer      { CHARACTERS } ]
    [                           { MODULES    } ]
    [,PROGRAM COLLATING SEQUENCE IS alphabet-name].

SPECIAL-NAMES.

    [ ,{ SYSIN  } IS mnemonic-name-1 ]
    [  { SYSOUT }                    ]
    [ ,{ TAB }    IS mnemonic-name-2 ]

    [                Ø                                                          ]
    [              { . }                                                        ]
    [ SWITCH       { . }   [IS mnemonic-name]  ON STATUS IS condition-name-1    ]
    [              { . }                                                        ]
    [              { 7 }                                                        ]

                   [OFF STATUS IS condition-name-2]

    [ , alphabet-name IS                                                          ]
    [                                                                             ]
    [          STANDARD-1                                                         ]
    [          NATIVE                                                             ]
    [          implementor-name                                              ...  ]
    [                      [{ THROUGH                                   }]        ]
    [          literal-1   [{ THRU         literal-2                    }]        ]
    [                      [{ ALSO literal-3 [, ALSO literal-4] ...      }]        ]
    [                                                                             ]
    [            [            [{ THROUGH                              }] ]        ]
    [            [ literal-5  [{ THRU         literal-6               }] ]        ]
    [            [            [{ ALSO literal-7 [, ALSO literal-8]     }] ]...     ]

     [,CURRENCY SIGN IS literal-9]
     [,DECIMAL-POINT IS COMMA]
     [,CURSOR IS data-name-1]
     [,CONSOLE IS CRT]          .
```

```
[INPUT-OUTPUT SECTION.

 FILE-CONTROL.

      {file-control-entry}...  ].

[I-O-CONTROL.

    [ ; RERUN [ON {file-name-1        }]
                  {implementor-name   }

                     {{[END OF]{REEL}       }       OF file-name-2}]   D
                     {{        {UNIT}       }                     }
               EVERY {{  integer-1 RECORDS  }                     }
                     {integer-2 CLOCK-UNITS                       }
                     {condition-name                              }

[ ; SAME    [RECORD    ]  AREA FOR file-name-3  {,file-name-4}... ] ...
            [SORT      ]
            [SORT-MERGE]

[ ; MULTIPLE FILE TAPE CONTAINS file-name-5     [POSITION integer-3]
                                                                      D
      [, file-name-6   [POSITION integer-4]   ]        ...] ...]   .
```

GENERAL FORMAT FOR FILE-CONTROL ENTRY

Sequential SELECT:

```
     SELECT file-name     [ OPTIONAL ] file-name

     ASSIGN TO    external-file-name-literal     [,{external-file-name-literal}]
                  file-identifier                [ {file-identifier           }]

    [; RESERVE integer-1     [{AREA }]]                    D
    [                        [{AREAS}]]

     ;ORGANIZATION IS [{SEQUENTIAL     }]
                      [{LINE SEQUENTIAL}]

     [;ACCESS MODE IS SEQUENTIAL]

     [;FILE STATUS IS data-name]  .
```

Relative Select:

```
     SELECT file-name

     ASSIGN TO     {external-file-name-literal}  [,{external-file-name-literal}]
                   {file-identifier           }  [ {file-identifier           }]

    [; RESERVE integer-1     [{AREA }]]                    D
    [                        [{AREAS}]]

     ORGANIZATION IS RELATIVE

    [                   {  SEQUENTIAL      [,RELATIVE KEY IS data-name]}]
    [;ACCESS MODE IS    {{RANDOM }          ,RELATIVE KEY IS data-name }]
    [                   {{DYNAMIC}                                     }]

     [;FILE STATUS IS data-name]  .
```

Indexed Select:

```
     SELECT file-name

     ASSIGN TO     {external-file-name-literal}  [,{external-file-name-literal}]
                   {file-identifier           }  [ {file-identifier           }]

    [; RESERVE integer-1     [{AREA }]]                    D
    [                        [{AREAS}]]
```

```
     ;ORGANIZATION IS INDEXED

    ┌                    ┌ SEQUENTIAL ┐ ┐
    │;ACCESS MODE IS    {  RANDOM      }│
    └                    └ DYNAMIC    ┘ ┘

     ;RECORD KEY IS data-name-1

     [; ALTERNATE RECORD KEY IS data-name-2 [WITH DUPLICATES ]  ]   ...

     [;FILE STATUS IS data-name-3]  .


Sort or Merge Select:

     SELECT file-name

     ASSIGN TO { external-file-name-literal }  ...  .
               { file-identifier            }
```

GENERAL FORMAT FOR THE DATA DIVISION

```
      DATA DIVISION.
   [  FILE SECTION.
    [ FD file-name

          [; BLOCK CONTAINS [integer-1 TO] integer-2  { RECORDS    }]
          [                                           { CHARACTERS }]   D

          [; RECORD CONTAINS  [integer-1  TO] integer-2  CHARACTERS]    D
           ; LABEL { RECORD IS   } { STANDARD }                         D
                   { RECORDS ARE } { OMITTED  }
     [; VALUE OF data-name-1 IS {data-name-2}
     [                          {literal-1  }
          [, data-name-3 IS{data-name-4}]       ]
          [                {literal-2  }]   ... ]             D

     [; DATA  { RECORD IS   }  data-name-3 [, data-name-4]...]        D
     [        { RECORDS ARE }                                ]


     [; LINAGE IS   {data-name-5}   LINES  [, WITH FOOTING AT  {data-name-6}]
     [              {integer-5  }          [                   {integer-6  }]
     [

          [, LINES AT TOP {data-name-7}][, LINES AT BOTTOM  {data-name-8}]]
          [               {integer-7  }][                   {integer-8  }]]

     [; CODE-SET IS alphabet-name]   .                        D

 [record-description-entry] ... ]...
 [SD file-name
        [                                                                 ]
        [; RECORD CONTAINS   [integer-1 TO]     integer-2  CHARACTERS     ]
        [                                                                 ]   D
        [; DATA{RECORD   IS }   data-name-1       [, data-name-2] ...     ]  .
        [      {RECORDS  ARE}                                             ]
  {record-description-entry}    ... ]  ... ]

        [ WORKING-STORAGE SECTION                        ]
        [ [77-level-description-entry]                   ]
        [ [record-description-entry  ]    ...            ]

        [ LINKAGE SECTION                                ]
        [ [77-level-description-entry]                   ]
        [ [record-description-entry  ]  ...              ]

        [ COMMUNICATION SECTION                                ]
        [                                                      ]
        [   [communication-description-entry]                  ]
        [   [record-description-entry     ...]  ...            ]
```

GENERAL FORMAT FOR DATA DESCRIPTION ENTRY

Format 1:

```
level-number {data-name-1}
             {FILLER     }
     [;REDEFINES data-name-2]

     [;{PICTURE} IS picture-string]
     [ {PIC    }                  ]

     [         {COMPUTATIONAL  } ]
     [         {COMP           } ]
     [;USAGE IS{COMPUTATIONAL-3} ]
     [         {COMP-3         } ]
     [         {DISPLAY        } ]
     [         {INDEX          } ]

     [[; SIGN IS]  {LEADING }  [SEPARATE CHARACTER]]
     [             {TRAILING}                      ]

     [ ; OCCURS {integer-1  TO integer-2  TIMES  DEPENDING ON data-name-3} ]
     [          {integer-2  TIMES                                        } ]
       [{ASCENDING } KEY IS data-name-4     [,data-name-5]] ... ...
       [{DESCENDING}                                     ]
           [INDEXED BY index-name-1    [, index-name-2]  ...]
                                                               D
     [;{SYNCHRONIZED}    {LEFT }]
     [ {SYNC        }    {RIGHT}]

     [;{JUSTIFIED}        RIGHT  ]
     [ {JUST     }               ]
     [;BLANK WHEN ZERO]
     [;VALUE IS literal] .
```

Format 2:

```
66 data-name-1; RENAMES data-name-2  [{THROUGH}  data-name-3]
                                     [{THRU   }             ]
```

Format 3:

```
88 condition-name; {VALUE IS  } literal-1 [{THROUGH}    literal-2]
                   {VALUES ARE}           [{THRU   }             ]

    [, literal-3  [{THROUGH}  literal-4] ]  ... .
    [             [{THRU   }           ] ]
```

GENERAL FORMAT FOR COMMUNICATION DESCRIPTION ENTRY

FORMAT 1:

```
CD cd-name;
                          [[; SYMBOLIC QUEUE IS data-name-1]
                                [; SYMBOLIC SUB-QUEUE-1 IS data-name-2]
                                [; SYMBOLIC SUB-QUEUE-2 IS data-name-3]
                                [; SYMBOLIC SUB-QUEUE-3 IS data-name-4]
                                [; MESSAGE DATE IS data-name-5]
FOR  [INITIAL ] INPUT           [; MESSAGE TIME IS data-name-6]
                                [; SYMBOLIC SOURCE IS data-name-7]
                                [; TEXT LENGTH IS data-name-8]
                                [; END KEY IS data-name-9]
                                [; STATUS KEY IS data-name-10]
                                [; MESSAGE COUNT IS data-name-11]]
                          [data-name-1, data-name-2, ..., data-name-11]]
```

FORMAT 2:

```
CD cd-name;  FOR OUTPUT
    [; DESTINATION COUNT IS data-name-1
    [; TEXT LENGTH IS data-name-2]
    [; STATUS KEY IS data-name-3]
   [[; DESTINATION TABLE OCCURS integer-2 TIMES
         [[; INDEXED BY index-name-1    [, index-name-2]  ...]]
    [; ERROR KEY IS data-name-4]
    [; SYMBOLIC DESTINATION IS data-name-4]
```

GENERAL FORMAT FOR PROCEDURE DIVISION

Declarative format:

```
    PROCEDURE DIVISION [USING data-name-1   [, data-name-2] ...].

    DECLARATIVES.
   {section-name SECTION [segment-number]. declarative-sentence
     [paragraph-name.  [sentence] ... ] ... } ...

    END DECLARATIVES.
   {section-name SECTION   [segment-number]  .
     [paragraph-name. [sentence] ... ] ... } ...
```

Non-declarative format:

```
    PROCEDURE DIVISION [ USING  data-name-1   [,data-name-2]   ... ] .
     {paragraph-name. [sentence] ...} ...
```

GENERAL FORMAT FOR VERBS

```
ACCEPT  dataname-1 [ AT  {data-name-2} ]   FROM CRT
                         {literal-1  }

ACCEPT identifier   [FROM CONSOLE]

                         {DATE}
ACCEPT identifier FROM   {DAY }
                         {TIME}

ACCEPT cd-name MESSAGE COUNT                               D

ADD {identifier-1} [identifier-2]...  TO  identifier  [ROUNDED]
    {literal-1   } [literal-2   ]

                [; ON SIZE ERROR imperative-statement]
ADD {identifier-1}  {identifier-2} ;[identifier-3    ]
    {literal-1   }  {literal-2   }  [literal-3    ...]

                GIVING identifier  [ROUNDED]

                [; ON SIZE ERROR  imperative-statement]

ADD { CORRESPONDING } identifier-1  TO  identifier-2  [ROUNDED]
    { CORR          }

ALTER { procedure-name-1 [TO PROCEED TO] procedure-name-2 }       ...

CALL {identifier-1}  USING  data-name-1    [, data-name-2]  ...
     {literal-1   }

CANCEL { identifier-1 }  [{,identifier-2}]      ...
       { literal-1    }  [{,literal-2   }]

                     D            D                        D
CLOSE file-name {REEL}      [WITH LOCK]  [,file-name  [WITH LOCK] ]   ...
                {UNIT}

                  [ {REEL}   [WITH NO REWIND] ]           D
                  [ {UNIT}   [FOR REMOVAL   ] ]
CLOSE file-name-1 [                           ]
                  [  WITH    {NO REWIND}      ]
                  [          {LOCK     }      ]

                  [[ {REEL}   [WITH NO REWIND] ]]
                  [[ {UNIT}   [FOR REMOVAL   ] ]]
 , file-name-2    [[                           ]]         ...
                  [[  WITH    {NO REWIND}      ]]
                  [[          {LOCK     }      ]]
```

```
                                        D                              D
CLOSE     file-name-1     [WITH LOCK]  [, file-name-2   [WITH LOCK]  ]  ...

COMPUTE   identifier-1    [ ROUNDED ]  [, identifier-2 [ROUNDED]  ]  ...

     = arithmetic-expression [; ON SIZE ERROR imperative-statement]


DELETE  file-name  RECORD  [; INVALID KEY  imperative-statement]   D

DISABLE { INPUT  }[TERMINAL]   cd-name WITH KEY   {identifier-1}   D
        { OUTPUT }                                {literal-1   }


DISPLAY {identifier-1} , {identifier-2}   ...  UPON CONSOLE
        {literal-1   }   {literal-2   }

DISPLAY {data-name-1} AT {data-name-2}   UPON {CRT      }
        {literal-3  }    {literal-4  }        {CRT-UNDER}

DIVIDE {identifier-1}   INTO  identifier-2   [ROUNDED]
       {literal-1   }


          [, identifier-3     [ROUNDED]  ]  ...
          [:ON SIZE ERROR     imperative-statement]


DIVIDE {identifier-1}  {INTO}  {identifier-2} GIVING identifier-3  [ROUNDED]
       {literal-1   }  {BY  }  {literal-2   }

          REMAINDER identifier-4 [;ON SIZE ERROR  imperative-statement]

                                                                      D

ENABLE  { INPUT   [TERMINAL] }  cd-name WITH KEY   {identifier-1}
        { OUTPUT             }                     {literal-1   }

ENTER  language-name  [routine-name].

EXIT   [PROGRAM].

GO TO[procedure-name].

GO TO procedure-name-1  {, procedure-name-2}...

     DEPENDING ON identifier

IF condition;  {statement-1  }    [; ELSE statement-2      ]
               {NEXT SENTENCE}    [; ELSE NEXT SENTENCE    ]
```

```
INSPECT  identifier-1 TALLYING  tally-clause (as follows)
   ⎧                          ⎧ ⎧ ALL                          ⎫⎫
   ⎪ identifier-2 FOR         ⎨,⎨ LEADING       ⎧identifier-3⎫ ⎬⎬
   ⎪                          ⎩ ⎩ CHARACTERS    ⎩literal-2   ⎭ ⎭⎪
   ⎨                                                    ⎫       ⎬- (tally-clause)
   ⎪   ⎡⎧BEFORE⎫             ⎧identifier-4⎫⎤  ⎬       ⎪
   ⎪   ⎢⎩AFTER ⎭   INITIAL   ⎩literal-3   ⎭⎥  ⎪       ⎪
   ⎩   ⎣                                    ⎦  ⎭       ⎭

INSPECT identifier-1 REPLACING   replacing-clause (as follows)
⎧                                                                ⎫
⎪    CHARACTERS BY   ⎧identifier-6⎫                              ⎪
⎪                    ⎩literal-4   ⎭                              ⎪
⎪                                                                ⎪
⎪ ⎧      ⎧ALL    ⎫                                               ⎪
⎨ ⎨[,]   ⎨LEADING⎬  ,  ⎧identifier-5⎫   BY   ⎧identifier-6⎫     ⎬- (replacing clause)
⎪ ⎩      ⎩FIRST  ⎭     ⎩literal-3   ⎭        ⎩literal-4   ⎭     ⎪
⎪                                                                ⎪
⎪      ⎧BEFORE⎫   INITIAL   ⎧identifier-7⎫  ⎫                    ⎪
⎪      ⎩AFTER ⎭             ⎩literal-5   ⎭  ⎬                    ⎪
⎩                                           ⎭                    ⎭

INSPECT identifier TALLYING tally-clause REPLACING replacing-clause

MERGE file-name-1 ON ⎧ASCENDING ⎫KEY data-name-1   [, data-name-2]  ...
                     ⎩DESCENDING⎭

                  ⎡ ON ⎧ASCENDING ⎫KEY daya-name-3   [, data-name-4]   ...⎤ ...
                  ⎣    ⎩DESCENDING⎭                                       ⎦

     [COLLATING SEQUENCE IS alphabet-name]

     USING file-name-2, file-name-3 [, file-name-4] ...
    ⎧                                         ⎡⎧THROUGH⎫                  ⎤⎫
    ⎪OUTPUT PROCEDURE IS section-name-1       ⎣⎩THRU   ⎭   section-name-2 ⎦⎪
    ⎨                                                                      ⎬
    ⎪GIVING file-name-5                                                    ⎪
    ⎩                                                                      ⎭

MOVE ⎧identifier-1⎫  TO identifier-2    [,identifier-3] ...
     ⎩literal-1   ⎭

MOVE  ⎧CORRESPONDING⎫ identifier-1 TO identifier-2
      ⎩CORR         ⎭
```

```
MULTIPLY {identifier-1} BY identifier-2  [ROUNDED]
         {literal-1   }

     [, identifier-3 [ROUNDED]   ...  [; ON SIZE ERROR imperative-statement]

MULTIPLY {identifier-1} BY{identifier-2} GIVING identifier-3  [ROUNDED]
         {literal-1   }   {literal-2   }

       [, identifier-4 [ROUNDED]   ...

       [; ON SIZE ERROR  imperative-statement]

        INPUT file-name-1 [REVERSED       ][,file-name-2  [REVERSED       ]]...     D
                          [WITH NO REWIND ][              [WITH NO REWIND ]]
     {
OPEN {  OUTPUT file-name-3  [WITH NO REWIND]  ,file-name-4 [WITH NO REWIND] ] ...    ...
     {  I-O file-name-5    [, file-name-6] ...
     {  EXTEND  file-name-7     [, file-name-8]  ...

PERFORM  procedure-name-1[{THROUGH} procedure-name-2]
                         [{THRU   }                 ]

                                                    {identifier-3}
PERFORM  perform-limits [VARYING {identifier-2} FROM {index-name-2}
                        [        {index-name-1}      {literal-1   }

                         BY      {identifier-4}] UNTIL  condition-1
                                 {literal-2   }]

                                                      identifier-6
                        [AFTER   {identifier-5} FROM  index-name-4
                        [        {index-name-3}       literal-3

                         BY      {identifier-7} UNTIL  condition-2
                                 {literal-4   }

                                                      identifier-9
                        [AFTER   {identifier-8} FROM  index-name-6
                        [        {index-name-5}       literal-5   ]

                         BY      {identifier  } UNTIL  condition-3]]
                                 {literal-6   }

READ file-name    [NEXT]   RECORD   [INTO identifier]

        [;AT END imperative-statement]

READ file-name  RECORD   [INTO identifier]  [;KEY IS data-name]

        [;INVALID KEY imperative-statement]

                                                                                  D
```

```
                 (MESSAGE)
RECEIVE cd-name  {       }  INTO identifier-1  [; NO DATA imperative-statement]
                 (SEGMENT)

RELEASE record-name  [FROM identifier]

RETURN file-name RECORD  [INTO identifier]   ; AT END imperative-statement

REWRITE  record-name  [FROM identifier]

         [;INVALID KEY imperative-statement]

SEARCH  identifier-1    [VARYING  {identifier-2}]
                        [         {index-name-1}]

      [;  AT END  imperative-statement-1]

       ; WHEN condition-1      {imperative-statement-2}
                               {NEXT SENTENCE         }

      [; WHEN condition-2      {imperative-statement-3}]
      [                        {NEXT SENTENCE         }]

SEARCH ALL  identifier-1  [; AT END imperative-statement-1]

                   (                                   identifier-3            )
                   (  data-name-1      {IS EQUAL TO}   literal-1               )
      ; WHEN       {                   {IS =       }   arithmetic-expression-1 }
                   (                                                           )
                   (  condition-name-1                                         )

    [              (                                   identifier-4            )]
    [              (  data-name-2      {IS EQUAL TO}   literal-2               )]
    [  AND         {                   {IS =       }   arithmetic-expression-2 }] ...
    [              (                                                           )]
    [              (  condition-name-2                                         )]

    {imperative-statement-2}
    {NEXT SENTENCE         }

SEND cd-name FROM  identifier-1

                                          (WITH identifier-2)
SEND cd-name   [FROM identifier-1]        {WITH ESI         }
                                          {WITH EMI         }
                                          (WITH EGI         )

   [                    ( {identifier-3}        [LINE ] )]
   [                    ( {integer     }        [LINES] )]
   [ BEFORE  ADVANCING  {                               }]
   [ AFTER              ( {mnemonic-name}               )]
   [                    ( {PAGE         }               )]
```

```
                                                             (identifier-3 )
SET  (identifier-1)  [identifier-2]    ...      TO           (index-name-3 )
     (index-name-1)  [index-name-2]                          (integer-1    )

                                                             (identifier-4 )
SET  (index-name-4)  [, index-name-5]      (UP BY  )         (integer-2    )
     (identifier-5)  [  identifier-6] ...  (DOWN BY)         (index-name-6 )

SORT  file-name-1  ON (ASCENDING )  KEY data-name-1  [, data-name-2] ...
                      (DESCENDING)

                   [ON (ASCENDING )  KEY data-name-3  [, data-name-4]] ...  ...
                   [   (DESCENDING)                                 ]

   [COLLATING SEQUENCE IS alphabet-name]

   (INPUT PROCEDURE IS  section-name-1 [(THROUGH)  section-name-2] )
   (                                   [(THRU   )                ] )
   (USING file-name-2    ,[file-name-3] ...                        )

   (OUTPUT PROCEDURE IS section-name-3 [ (THROUGH)   section-name-4] )
   (                                   [ (THRU   )                 ] )
   (GIVING file-name-4                                               )

                   [       ( IS EQUAL TO        )                    ]
                   [       ( IS =               )                    ]
START   file-name  [ KEY   ( IS GREATER than    )          data-name ]
                   [       ( IS >               )                    ]
                   [       ( IS NOT LESS THAN   )                    ]
                   [       ( IS NOT <           )                    ]
            [;INVALID  KEY  imperative-statement]

STOP     (RUN    )
         (literal)

                                                                 (identifier-3)
STRING   (identifier-1) [,   (identifier-2)] ... DELIMITED BY    (literal-3   )
         (literal-1   ) [    (literal-2   )]                     (SIZE        )

  [                                                               (identifier-6)]
  [ ,    (identifier-4) [,   (identifier-5)] ...  DELIMITED  BY   (literal-6   )] .
  [      (literal-4   ) [    (literal-5   )]                      (SIZE        )]

   INTO   identifier-7       [WITH POINTER  identifier-8]

   [, ON OVERFLOW imperative-statement

SUBTRACT (identifer-1) [, (identifier-2)]...     FROM identifier-m    [ROUNDED]
         (literal-1  ) [  (literal-2   )]

           [, identifier-n [ROUNDED] ]  ...

           [; ON SIZE ERROR  imperative-statement]
```

```
SUBTRACT {identifier-1} , {identifier-2} ...  FROM {identifier-m}
         {literal-1   }   {literal-2   }           {literal-m   }

        GIVING  identifier -n   [ROUNDED] [, identifier-o      [ROUNDED]] ...

        [; ON SIZE ERROR  imperative-statement]

UNSTRING identifier-1
    [DELIMITED  BY [ALL]     {identifier-2} [, OR [ALL]  {identifier-3}]...]
                             {literal-1   }              {literal-2   }

      INTO identifier-4   [, DELIMITER IN identifier-5]  [, COUNT IN identifier-6]

[, identifier-7  [, DELIMITER IN identifier-8][, COUNT IN identifier-9]] ...

     [WITH POINTER identifier-10] [TALLYING IN identifier-11]

     [; ON OVERFLOW imperative-statement]

                                                  (file-name-1) [ , file-name-2 ] ...
                                                  (INPUT      )
USE AFTER STANDARD {EXCEPTION} PROCEDURE ON      (OUTPUT     )
                   {ERROR    }                    (I-O        )
                                                  (EXTEND     )

                          (cd-name-1                               )
                          ([ALL REFERENCES OF] identifier-1        )
USE FOR DEBUGGING ON      (file-name-1                             )
                          (procedure-name-1                        )
                          (ALL PROCEDURES                          )

    [      cd-name-2                              ]
    [      [ALL REFERENCES OF] identifier-2       ]
    [ ,    file-name-2                            ]        ...  .
    [      procedure-name-2                       ]
    [      ALL PROCEDURES                         ]

WRITE record-name   [FROM  identifier-1 ]
       [                         ( integer          {LINE } ) ]
       [ {BEFORE}  ADVANCING     ( identifier-2     {LINES} ) ]
       [ {AFTER }                ( {PAGE}                   ) ]
       [                         ( {TAB }                   ) ]
        [; AT       {END-OF-PAGE}    imperative statement]
                    {EOP        }

WRITE   record-name  FROM  identifier

          [;INVALID KEY imperative-statement]
```

GENERAL FORM FOR COPY STATEMENT

```
COPY     "text-name"
```

GENERAL FORMAT FOR CONDITIONS

Relation condition:

```
                          /IS [NOT] GREATER THAN\
 (identifier-1           )(IS [NOT] LESS THAN    )  (identifier-2           )
 )literal-1              )(IS [NOT] EQUAL to     (  )literal-2              )
 )arithmetic-expression-1(IS [NOT] >             (  )arithmetic-expression-2(
 (index-name-1           )(IS [NOT] <            )  (index-name-2           )
                          \IS [NOT] =           /
```

Class Condition:

```
identifier IS [NOT]   {NUMERIC   }
                      {ALPHABETIC}
```

Sign Condition:

```
                                    {POSITIVE}
arithmetic-expression IS [NOT]      {NEGATIVE}
                                    {ZERO    }
```

Condition-name Condition:

condition-name

Switch-status Condition:

condition-name

Negated Simple Condition:

NOT simple-condition

Combined Condition:

```
condition    { {AND} condition }    ...
             { {OR }           }
```

Abreviated Combined Relation Condition:

```
relation-condition{ {AND} NOT  [relational-operator]       object }...
                  { {OR }                                         }
```

MISCELLANEOUS FORMATS

QUALIFICATION:

```
{data-name-1   }  [{OF}   data-name-2 ]  ...
{condition-name}  [{IN}               ]

paragraph-name   [{OF}  section-name ]
                 [{IN}               ]

text-name   [{OF}  library-name ]
            [{IN}               ]
```

SUBSCRIPTING:

```
{data-name     }    (subscript-1   [, subscript-2  [, subscript-3]] )
{condition-name}
```

INDEXING:

```
{data-name     } (  {index-name-1  [± literal-2]}
{condition-name}    {literal-1                  }
  [  {index-name-2  [± literal-4]}  [   {index-name-3  [± literal-6]} ]]
  [, {literal-3                  }  [,  {literal-5                  } ]] )
```

IDENTIFIER: FORMAT 1

```
data-name-1  {OF}  data-name-2    ... [(subscript-1    [, subscript-2
             {IN}

     [, subscript-3]]    )]
```

IDENTIFIER: FORMAT 2

```
data-name-1  [{OF}  data-name-2 ]  ... [(   {index-name-1   [ ± literal-2]}
             [{IN}              ]      [    {literal-1                   }
    , [{index-name-2   ± literal-4 }  , [{index-name-3  [ ± literal-6]}]] )]
      [{literal-3                  }    [{literal-5                   }]]  ]
```

apple computer

20525 Mariani Avenue
Cupertino, California 95014
(408) 996-1010
TLX 171-576
030-0483-A

Photographs courtesy of Vorpal Gallery, San Francisco, New York, Laguna Beach, CA.