SPERRY UNIVAC
COMPUTER SYSTEMS

COBOL PROGRAM DEVELOPMENT SUBSYSTEM (CODE) AND TEST FILE GENERATOR (TFG) REFERENCE CARD

LOGGING ON

```
/LOGON userid,accntno[,password] [,PRIORITY=n]
  [,MSG= {F} [H][,BUFFER=integer] [TIME=sec-integer]
         {C}
```

```
%C E223 LOGON ACCEPTED FROM LINE nn AT time
  ON date, TSN=nnnn
```

Comments

User logs on

System replies

CREATING (OR ERASING) A CODE/TFG PROJECT

A given project is created only once. It exists until erased.

```
/EXEC COBOL
% C P500 LOADING.
  ENTERING INITIALIZATION MODE
  ENTER PROGRAM-ID OR #HALT.
```

User enters Execute command
System replies

Initializing For Series 70 COBOL

```
*'program-id'

  INITIALIZED FOR SERIES 70 COBOL; TO ACCESS
  /DO program-id
```

User enters 'program name' in single quotes.
System replies
User enters

Initializing For ANS COBOL

```
*program-id

  NON-NUMERIC LITERALS WILL BE IN ' OR "?
  *' or *"

  INITIALIZED FOR AMERICAN STANDARD COBOL;
  TO ACCESS
  /DO program-id
```

User enters program name without quotes
System replies
User enters

System replies

User enters

Erasing An Existing CODE/TFG Program

```
*program-id
  NAME EXISTS; O=OVERWRITE/R=RESTART/
  E=ERASE/Q=QUITS
*E
  ERASED ALL REFERENCES
  ENTER PROGRAM-ID OR #HALT
*HALT or another program name
```

User enters program name
System replies

User enters E
System replies
System replies
User enters

EDITING A PROGRAM IN CODE

/DO program-id Command:

```
/DO program-id
% C P001 -DLL V-15.
  ENTERING EDIT MODE
  DEFINITION FILE ISN'T THERE
  SELECT::S=SYSTEM/U=USER/N=NONE/
  PROGRAM NAME
*S or *U or *N or *program-id
*P001.000        #SYNTAX
#AT        #BACKUP     #COMPILE    #DEFINE
#DELETE    #DUPLICATE  #EXECUTE    #FIND
#GET       #HALT       #LIST       #LOAD
#LOCATE    #MESSAGES   #NOTE       #PRINT
#QUESTION  #REPLACE    #SAVE       #SET UP
#SHORTHAND #STEP       #SUMMARY    #SYNTAX
#TEXT      #UNDEFINE   #UNITS      #VERIFY
```

User enters
System replies
System continues
System continues
System continues

User enters S, U, or N
System replies and user enters command
System replies with list of CODE commands. A complete list of the CODE commands, and the syntax of each command is appended to this card.

EA-036-0-00 (6/74)

EDITING A PROGRAM IN CODE (Cont'd)

Source Statements and Commands:	Comments
* P001.000 #AT I	System continues User writes the Identification Division of his program
* I001.000 user program (IDENTIFICATION DIVISION) * I002.000 * I003.000	
S I00n.000 #AT E	User writes the Environment Division of his program
* E001.000 user program (ENVIRONMENT DIVISION) * E002.000 * E003.000	
S E00n.000 #AT D	User writes the Data Division of his program
* D001.000 user program (DATA DIVISION) * D002.000 * D003.000	
S D00n.000 #AT P	User writes the Procedure Division of his program
* P001.000 user program (PROCEDURE DIVISION) * P002.000 * P003.000	

PROGRAM PROCESSING

Compiling the Program	
*P00n.000 #COMPILE LIST RETURN	User enters Compile command as last statement in program
SAVE LATEST DEFINITIONS ?Y=YES/N=NO *Y or *N	System replies Definitions created with #DEFINE in this session are saved or not saved
ENTERING FOREGROUND JOB %P500 LOADING	System replies Compilation loading as foreground job
32A0 COMPILATION INITIATED (BGCOB VERSION=nnn) 32AA COMPILATION COMPLETED WITH SERIOUS ERRORS %EB001 SPOOLOUT INITIATED FOR TSN=nnnn ID= nnnnnnnn % PRINT FILE=nnnnn	System replies Listing (TSN=nnnn) produced on system printer Print command
Debugging the Compilation	
%P500 LOADING	Compiler loads diagnostic information into CODE error file
CREATED DIAGNOSTIC FILE %P001 – DLL V-09 ENTERING EDIT MODE *P001.000 #SUMMARY	System replies and user enters command
T=nnn E.=nnn M=nnnn	System prints the diagnostic message for specific errors
B1006#001 B1006#002 B1006#003 . #Diagnostic Messages . B1006#00n	

PROGRAM PROCESSING (Cont'd)

Debugging the Compilation (Cont'd)	Comments
* P001.000 #AT D4.3 . Corrections . .	User starts to correct his program
* P00n.000 #COMPILE LIST RETURN TEST	User recompiles program and prepares it for use by TFG
ENTERING FOREGOUND JOB %P500 LOADING 32A0 COMPILATION INITIATED (BGCOB VERSION=nnn) 32AA COMPILATION COMPLETED %EB001 SPOOLOUT INITIATED FOR TSN=nnnn ID= nnnnnnnn % PRINT FILE=nnnn %P500 LOADING . . .	
*P001.000 #HALT	User issues HALT to return control to operating system

TEST FILE GENERATOR (TFG)

/DO TFGprogram-id %P001 - DLL V-09 (COBOL 5) TEST FILE GENERATOR LOADED VERSION # 001 SPECFILE CREATION DATE nn/nn/nn VERSION # 002 ENTER OUTPUT MODE OR ?	TFG invoked by user System replies System asks how spec lines are desired
*B	User enters B for both Data names and Nos. displayed
ENTER OPEN COMMAND *#OPEN SAMPLE-FILE	System replies User names file
Specifying Rules For Elementary Items	
FILE 'SAMPLE-FILE' OPENED $001 SAMPLE-RECORD-1	System opens file System prompts with record names
001.100 FIRST ITEM	System prompts with item name
*VALUE IS 'ABC' 002.100 SECOND-ITEM	User gives value for item System prompts with next record
*VALUE IS RANDOM BETWEEN 100 AND 500 FOR 50 RECORDS . . . $002 SAMPLE-RECORD-2	User gives rule for second item System prompts with next record
002.100 ITEM-X	System prompts with data name
*VALUE IS 1, 2, 3, 4 REPEAT . . .	User gives series of values

General Format of Rules For Elementary Items

VALUE[IS] { constant | RANDOM | v-1 . . . v-n[REPEAT] | FIRST-NAME | LAST-NAME | NAME | SPACE | MONTH | WEEK-DAY | ZERO | HIGH-VALUE | LOW-VALUE | arithmetic expression | data-name | NULL }

constant may be a literal, a hexadecimal constant (X'1F'), or a (B'01101011')
v-1 to v-n may be any of the parameters legal for VALUE

TEST FILE GENERATOR (TFG) (Cont'd)

General Format of Rules For Elementary Items (Cont'd)	Comments
(INCREMENT / INC / DECREMENT / DEC) [IS] (numeric-literal / RANDOM / arithmetic expression / data-name) (MAXIMUM / MAX / MINIMUM / MIN) [IS] (numerical-literal / data-name / arithmetic expression) BETWEEN (numerical-literal-1 / data-name-1 / arithmetic-expression-1) AND (numeric-literal-2 / data-name-2 / arithmetic-expression) FOR integer RECORDS	
Test File Generator Editing Commands #AT #DELETE #FIND #PRINT #STEP	These commands can be used while rules are being specified. The full syntax or the TFG commands is appended to this card
Specifying the Records in the File ENTERING RECORD CONTROL MODE	System comes to record control when rules have been given for all items or when the user types #AT &1
&001	System prompts with the first record control number
*1 $1 3 $2	User enters record group
&002	System prompts with the next record control number
*10 $2 100 &1	User calls for nesting of record groups
&003	System prompts with the next record control number
#GENERATE LIST	User decides to create the file with these rules
LOGGING OFF /LOGOFF %C E420 LOGOFF AT nnn ON nn/nn/nn FOR TSNnnnn %C E421 CPU TIME USE: nnnnnn.nnnn seconds	User logs off System replies

CODE COMMANDS	Operand Definitions
# (AT / A) ((+ / −) number [number] / unit number) [(STEP / S) step-number]	number= 1 to 99 units to be moved forward or backward unit number= alphanumeric division location step number= decimal increment to separate continuous unit nos.
# (BACKUP / B)	
# (COMPILE / C) [[(EXECUTE / E) / (LOAD / LO)]] [(LIBRARY / LIB)] [(LIST / LIS)] [(RETURN / R)] [TEST] [(NOFILE / N) / 'filename' / "filename"]	filename= user designated name of a cataloged COBOL source library

CODE COMMANDS (Cont'd)

	Operand Definitions
# (DEFINE / DEF) [(abbreviation [(AS / A) ('string' / "string")]) / ('letter' / "letter")]	Abbreviation= 1 to 5 characters used to identify a 'string' letter= letter with which a given abbreviation begins string= group of letters identified by a particular abbreviation
# (DELETE / DEL) [unit-number-1 [TO unit-number-2]]	unit-number-1= first unit no. of the range of units to be deleted unit-number-2= last unit no. of the range of units to be deleted
# (DUPLICATE / DU) unit-number-1 [TO unit-number-2]	unit-number-1= first unit no. of the range of units to be copied unit-number-2= last unit no. of the range of units to be copied
# (EXECUTE / E) [(RETURN / R)] [(LIBRARY / LIB)]	
# (FIND / F) [(FIRST / F) / (LAST / L) / (ALL / A)] ('string' / "string") [IN / (ON / O)] unit-number-1 [TO unit-number-2]	string= alphanumeric character configuration to be searched for unit-number-1. . . unit-number-2= first and last unit in a range, the contents of which are to be scanned for the string
# (GET / G) [record-number-1 [TO record-number-2]]	record-number-1. . . record-number-2= first and last unit in a range of records to be obtained from specified file filename= name of a file containing all or part of a COBOL source program in card-image format
(FROM / F) ('filename' / "filename" / TAPE / TA)	
# (HALT / H)	
# (LIST / LIS) [(CHANGES / C) / (ERRORS / E) / (SHORTHAND / SH) / (SOURCE / SO) / (SYNTAX / SY)]	
# (LOAD / LOA) [(LIBRARY / LIB)]	
# (LOCATE / LOC) ['string' / "string"]	string= alphanumeric character configuration identifying the unit for which the search is being conducted
# (MESSAGES / M) [unit-number]	unit-number= number of the unit to be searched for messages
# (NOTE / N) message-text	

CODE COMMANDS (Cont'd)

	Operand Definitions
# {PRINT / P} [[[[{FIRST / F} / {LAST / L} / {ALL / A}]] {'string' / "string"}] {IN / {ON / O}}] unit-number-1 [TO unit-number-2]]	string= alphanumeric character configuration being searched for unit-number-1. . . unit-number-2= first and last unit in a range of units, the contents of which are to be printed or unit-number where scanning should stop
# [PRINT +1]	
# {QUESTION / Q} message-text	message text= set of characters transmittable from a conversational terminal
# {REPLACE / R} [{FIRST / F} / {LAST / L} / {ALL / A}] {'string-1' / "string-1"} [{IN / {ON / O}}] unit-number-1 [TO unit-number-2]] {BY / B} { ' ' / " " / {'string-2' / "string-2"} }	string-1= alphanumeric character configuration to be replaced within the source text unit-number-1. . . unit-number-2= first and last unit number in a range of units being searched for string-1 string-2= alphanumeric character configuration designated to replace string-1
# {SAVE / SA} { [{CARDS / C} {'filename' / "filename"} / {TAPE / TA} / [{ISAM / I}]] / {LIBRARY / LIB} }	filename= name to be assigned to SAM or ISAM file into which source program is to be copied Note: filename must be different than the name of source program
# {SETUP / SE} [unit-number]	unit-number= absolute unit-number of REMARKS unit in which the SETUP list is specified
# {SHORTHAND / SH} [{ [{OFF / ON}] / {ERASE / E} / {SORT / S} }]	
# {STEP / ST} increment-value	increment-value= decimal number indicating the change in numerical increment between unit numbers
# {SUMMARY / SU} [[error-number OF] type]	error-number= number of message type type= type of message for which Summary information is being requested

CODE COMMANDS (Cont'd)

	Operand Definitions
# {SYNTAX / SY} [{# / #command-name / #letter[letter]}]	#= display a list of syntax of all CODE commands #command-name= display syntax of named command #letter= display syntax of all commands which begins with the specified letter
# {TEXT / T} [[error-number OF] type]	error-number= number of message type type= type of message for which the text is to be printed
# {UNDEFINE / UND} abbreviation	abbreviation= the abbreviation to be removed from Shorthand dictionary
# {UNITS / UNI} [[error-number OF] type]	error-number= number of error type type= type of message for which the source units are to be searched
# {VERIFY / V} [{NOFILE / N / 'filename' / "filename"}] [{LIST / LIS}] [{RETURN / R}]	filename= user-designated name of a cataloged COBOL source library

Note: The syntax of all the CODE commands can also be obtained at the terminal by issuing the #SY command with the # operand. It is also possible with this command, to obtain the syntax of a specific command (using #SY # command name), or a syntax list of all commands beginning with a particular letter or letters (using #SY # letter[letter]). To ensure that these commands are entered properly, the programmer should be thoroughly familiar with the Rules for Spacing, Command Verb Abbreviations, and Operand Specifications paragraphs under the heading Command Conventions in the CODE and TFG Reference Manual.

TFG COMMANDS

Editing Commands		Comments
#AT	{n [STEP .k] / n.J [STEP .k] / $n[STEP .k] / &n}	Allows programmer to set the specifications number to any number he desires.
#DELETE	{n [TO m] / n.J [TO n.k] / ALL / ALL $ / ALL $n / ALL & / &n to END}	Allows programmer to delete rule, record, or record control specifications.
#FIND	{data-name-1,..... ,data-name-n / record-name / ALL}	Allows programmer to display elementary items corresponding to any or all specification numbers.
#PRINT	{[ALL] n [TO m] / n.j [TO n.k] / [ALL] $n / &n [TO & m] / ALL}	Allows programmer to display pertinent information concerning attributes, specification numbers, and record controls.
#STEP	{.k / [1]}	Allows programmer to enter more than one rule specification for a given item.

TFG COMMANDS (Cont'd)

File Control Commands		Comments
#CLOSE	[LIST]	Causes TFG to terminate processing and return control to VMOS.
#GENERATE	[PRINT]	Generates a test file, and causes the file to be printed and returns control to VMOS.
#OPEN	file-name	Causes TFG to open the name of the file to be worked on.

COMMAND CONVENTIONS

Notation Conventions

The following notation is used in the formats of all CODE commands:

1. Fixed names are shown in capital letters. The term, fixed names, pertains to command names and certain operands.
2. Variable names are shown in lower case letters.
3. Optional items are enclosed in brackets, [ ].
4. Alternate items are enclosed in braces, { }.
5. Ellipses (. . .) following an operand indicate that the user may specify more than one operand of that type.

Rules for Spacing

The following spacing rules must be observed when typing commands:

1. Any number of spaces may appear between the # symbol and the command verb.
2. Any number of spaces may appear between command elements.
3. No spaces may appear within a command element unless the element is a string.

TFG Command Symbols

The following symbols are used in the command formats:

n	integer portion of first specification number	(1 to 3 digits)
m	integer portion of second specification number	(1 to 3 digits)
j	decimal portion of first specification number	(1 to 3 digits)
k	decimal portion of second specification number	(1 to 3 digits)
$n	record number	($ followed by 1 to 3 digits)
&m	record control number	(& followed by 1 to 3 digits)

TFG Command Conventions

The following conventions apply to the TFG commands:

1. The attributes of an elementary item are size, picture, and usage.
2. All data names must be fully qualified.
3. TFG follows the convention of the CODE subsystem of allowing the programmer to enter the minimum number of characters to distinguish a command. For example, the following are equivalent:

```
#P
#PR
#PRI
#PRIN
#PRINT
```