

**UNISYS**

A Series

CANDE

**Operations**

**Reference Manual**

Release 3.9.0

September 1991

Priced Item

Printed in U S America  
8600 1500-000



**UNISYS**

A Series

CANDE

**Operations**

**Reference Manual**

Copyright © 1991 Unisys Corporation.

All Rights Reserved.

Unisys is a registered trademark of Unisys Corporation.

Release 3.9.0

September 1991

Priced Item

Printed in U S America  
8600 1500-000

The names, places, and/or events used in this publication are not intended to correspond to any individual, group, or association existing, living, or otherwise. Any similarity or likeness of the names, places, and/or events with the names of any individual, living or otherwise, or that of any group or association is purely coincidental and unintentional.

**NO WARRANTIES OF ANY NATURE ARE EXTENDED BY THE DOCUMENT.** Any product and related material disclosed herein are only furnished pursuant and subject to the terms and conditions of a duly executed Program Product License or Agreement to purchase or lease equipment. The only warranties made by Unisys, if any, with respect to the products described in this document are set forth in such License or Agreement. Unisys cannot accept any financial or other responsibility that may be the result of your use of the information in this document or software material, including direct, indirect, special or consequential damages.

You should be very careful to ensure that the use of this information and/or software material complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

Correspondence regarding this publication may be forwarded using the Product Information card at the back of the manual, or may be addressed directly to Unisys, Product Information, 25725 Jeronimo Road, Mission Viejo, CA 92691.



# Page Status

Page	Issue
iii through iv	-010
v through vii	-000
viii	Blank
ix through xiii	-010
xiv	Blank
xv	-000
xvi	Blank
xvii	-010
xviii	Blank
1-1 through 1-2A	-010
1-2B	Blank
1-3 through 1-6	-000
1-7 through 1-10	-010
1-11 through 1-17	-000
1-18	Blank
2-1 through 2-9	-000
2-10	Blank
3-1 through 3-4	-000
3-4A through 3-4C	-010
3-4D	Blank
3-5 through 3-12	-000
3-13 through 3-20A	-010
3-20B	Blank
3-21 through 3-26	-000
3-27 through 3-50	-010
3-51 through 3-58	-000
3-58A through 3-58C	-010
3-58D	Blank
3-59 through 3-62	-000
3-63 through 3-74A	-010
3-74B	Blank
3-75 through 3-78A	-010
3-78B	Blank
3-79 through 3-84	-000
3-85 through 3-90C	-010
3-90D	Blank
3-91 through 3-120	-000
3-121 through 3-124C	-010
3-124D	Blank
3-125 through 3-130	-000
3-131 through 3-136A	-010
3-136B	Blank
3-137 through 3-146	-000

*continued*

## Page Status

---

*continued*

Page	Issue
3-147 through 3-152A	-010
3-152B	Blank
3-153 through 3-176	-000
3-177 through 3-180	-010
3-181 through 3-184	-000
3-185 through 3-190	-010
3-191 through 3-192	-000
3-193 through 3-194	-010
3-195 through 3-203	-000
3-204	Blank
4-1 through 4-56	-000
4-57 through 4-58A	-010
4-58B	Blank
4-59 through 4-78	-000
4-79 through 4-80A	-010
4-80B	Blank
4-81 through 4-88	-000
5-1 through 5-2	-010
A-1 through A-8	-000
Glossary-1 through 17	-000
Glossary-18	Blank
Bibliography-1 through 2	-000
Index-1 through 10	-010

Unisys uses an 11-digit document numbering system. The suffix of the document number (1234 5678-xyz) indicates the document level. The first digit of the suffix (x) designates a revision level; the second digit (y) designates an update level. For example, the first release of a document has a suffix of -000. A suffix of -130 designates the third update to revision 1. The third digit (z) is used to indicate an errata for a particular level and is not reflected in the page status summary.

# About This Manual

## Purpose

This manual documents the CANDE (Command and Edit) message control system (MCS), which provides generalized file preparation and updating capabilities in an interactive, terminal-oriented environment.

## Scope

This manual covers information about the capabilities, functions, and limitations of the CANDE MCS. Specific information about initiating tasks, manipulating work files, and interrogating the operating environment is provided.

The installation, configuration, and operational control of CANDE and its data communications (data comm) network are described in the *A Series CANDE Configuration Reference Manual*.

## Audience

This manual is designed for CANDE users who work with text and program files.

## Prerequisites

Familiarity with the A Series systems is recommended.

## How to Use This Document

As a reference tool, this manual can be read in any order. The Table of Contents and the Index provide quick page references to specific information.

Other documents that provide additional information about using CANDE are listed under "Related Product Information" at the end of this preface. All documents referred to in this manual are for A Series systems, unless otherwise noted.

Appendix A, "Understanding Railroad Diagrams," contains an explanation of the railroad diagrams used in this manual.

The Glossary defines terminology related to CANDE. Acronyms used in this manual are spelled out in the glossary.

The Bibliography contains a list of documents referenced in this manual.

## Organization

This manual is divided into the following five sections and one appendix. In addition, this manual contains a glossary, a bibliography, and an index.

### Section 1. General Information

The information in this section is designed to familiarize the user with the CANDE environment. It explains how to initiate the message control system (MCS) to establish a connection between CANDE and a physical station. It also covers log-on requirements, special CANDE files and operations, and the functions and limitations of the CANDE MCS.

### Section 2. Basic Constructs

This section defines the metatokens used to construct CANDE commands. The metatokens and their syntax diagrams are listed alphabetically for easy reference.

### Section 3. CANDE Commands

This section describes the CANDE commands used to initiate tasks and manipulate work files. The CANDE commands and their syntax diagrams are listed alphabetically for easy reference.

### Section 4. CANDE Control Commands

This section describes the CANDE control commands used to interrogate the operating environment. The control commands and their syntax diagrams are listed alphabetically for easy reference.

### Section 5. Functional Command Groupings

In this section, the CANDE commands are grouped by the types of functions they perform.

### Appendix A. Understanding Railroad Diagrams

This appendix describes how to read the railroad diagrams that are used for showing command syntax.

In addition, this manual contains a glossary, a bibliography, and an index.

## Related Products Information

### ***A Series CANDE Configuration Reference Manual (form 8600 1344)***

This manual describes the commands used to perform CANDE control functions and data communications network control functions. It also describes how to configure CANDE to meet the resource requirements of the installation. This manual is written for system administrators and operators.

### ***A Series Distributed Systems Service (DSS) Operations Guide (form 8600 0122)***

This guide describes the capabilities and features of DSS Services. It is intended for system operators, system administrators, and general computer users.

### ***A Series File Attributes Programming Reference Manual (form 8600 0064). Formerly A Series I/O Subsystem Programming Reference Manual***

This manual contains information about each file attribute and each direct I/O buffer attribute. The manual is written for programmers and operations personnel who need to understand the functionality of a given attribute. The *A Series I/O Subsystem Programming Guide* is a companion manual.

### ***A Series I/O Subsystem Programming Guide (form 8600 0056). Formerly A Series I/O Subsystem Programming Reference Manual***

This guide contains information about how to program for various types of peripheral files and how to program for interprocess communication, using port files. This guide is written for programmers who need to understand how to describe the characteristics of a file in a program. The *A Series File Attributes Programming Reference Manual* is a companion manual.

### ***A Series Printing Utilities Operations Guide (form 8600 0692)***

This guide describes how to use the Print System utilities: Backup Processor, SYSTEM/BACKUP, and LTTABLEGEN. This guide is written for programmers, system administrators, and interactive users of Menu-Assisted Resource Control (MARC) and CANDE who are familiar with the concepts and use of the Print System.

### ***A Series System Commands Operations Reference Manual (form 8600 0395)***

This manual gives a complete description of the system commands used to control system resources and work flow. This manual is written for systems operators and administrators.

### ***A Series Work Flow Language (WFL) Programming Reference Manual (form 8600 1047)***

This manual presents the complete syntax and semantics of WFL. WFL is used to construct jobs that compile or run programs written in other languages and that perform library maintenance such as copying files. This manual is written for individuals who have some experience with programming in a block-structured language such as ALGOL and who know how to create and edit files using CANDE or the Editor.



# Contents

About This Manual .....	v
<b>Section 1. General Information</b>	
Pseudostations and COMS Window Dialogs .....	1-1
User Identification and Logging On .....	1-1
Startup Files and Restart Files .....	1-3
Accesscodes and Recovery Files .....	1-4
ACCESSCODE Task Attribute .....	1-5
Multiple Commands .....	1-5
Family Substitution .....	1-6
Break Condition .....	1-6
Response to CANDE Commands .....	1-7
Saved Text Queue Manipulation .....	1-7
Input Queue Manipulation .....	1-7
Attributes of CANDE Files .....	1-9
Limitations of CANDE Files .....	1-9
Work Files .....	1-10
Recovery Files .....	1-10
Remote Files .....	1-12
Page Mode Operations .....	1-12
Requirements for Page Mode .....	1-12
Initiating Page Mode and Moving through the Work File .....	1-13
The Column Indicator .....	1-14
Entering Text and Commands in Page Mode .....	1-14
<b>Section 2. Basic Constructs</b>	
<b>Section 3. CANDE Commands</b>	
ACCESS .....	3-2
ADD .....	3-4
ALTER .....	3-4A
APASSWORD .....	3-5
BACKUPPROCESS .....	3-7
BIND .....	3-10
BYE .....	3-11
CHANGE .....	3-12
CHARGE .....	3-17
COMPILE .....	3-18
CONNECT .....	3-21
CONTINUE .....	3-23
CONVENTION .....	3-25

## Contents

---

<b>COPY or ADD</b> .....	3-27
Specifying File Attributes .....	3-34
File Transferring Services .....	3-40
A Series Native File Transfer (NFT) .....	3-41
Host Services File Transfer .....	3-41
File Transfer Protocol (FTP) .....	3-41
File Transfer, Access, and Management (FTAM) ..	3-42
Copying Files between Hosts .....	3-43
Restrictions on Copying Files between Hosts .....	3-44
NFT File Transfer Restrictions .....	3-44
Host Services File Transfer Restrictions .....	3-45
FTP File Transfer Restrictions .....	3-45
FTAM File Transfer Restrictions .....	3-46
Restarting Interrupted File Transfers .....	3-46
Examples .....	3-47
<b>DCSTATUS</b> .....	3-50
<b>DELETE</b> .....	3-52
<b>DESTNAME</b> .....	3-53
<b>DISCARD</b> .....	3-55
<b>DO</b> .....	3-56
<b>ESCAPE</b> .....	3-58A
<b>EXCLUDE</b> .....	3-59
<b>EXECUTE or RUN</b> .....	3-60
<b>FAMILY</b> .....	3-62
<b>FILES</b> .....	3-64
<b>FIND</b> .....	3-66
<b>FIX</b> .....	3-71
<b>GET or LOAD</b> .....	3-76
<b>HELLO</b> .....	3-78A
<b>INSERT</b> .....	3-80
<b>LANGUAGE</b> .....	3-82
<b>LFILES</b> .....	3-83
<b>LIST</b> .....	3-86
<b>LOAD</b> .....	3-90C
<b>LOG</b> .....	3-91
<b>MAKE</b> .....	3-92
<b>MARGIN</b> .....	3-95
<b>MARKID</b> .....	3-97
<b>MATCH</b> .....	3-101
<b>MCS</b> .....	3-105
<b>MERGE</b> .....	3-106
<b>MOVE</b> .....	3-108
<b>NEWS</b> .....	3-111
<b>NEXT</b> .....	3-112
<b>PAGE</b> .....	3-114
<b>PASSWORD</b> .....	3-116
<b>PDEF or PRINTDEFAULTS</b> .....	3-119
<b>PRINT</b> .....	3-121
<b>PRINTDEFAULTS</b> .....	3-125
<b>RANGE</b> .....	3-126
<b>RECOVER</b> .....	3-129
<b>REMOVE</b> .....	3-131



RENEW	3-134
REPLACE	3-135
RESEQ	3-140
RMERGE	3-142
RO	3-144
RUN	3-145
SAME	3-146
SAVE	3-147
SCHEDULE	3-149
SECURITY	3-153
SEQ	3-156
Single-Line Entry or Deletion	3-161
SO or RO	3-163
SPLIT	3-165
START	3-166
STATUS	3-172
STOP	3-173
TAB	3-174
TAPE	3-176
TERMINAL	3-178
TITLE	3-180
TYPE	3-181
UPDATE	3-183
UTILITY	3-185
VOID	3-190
WFL	3-192
WHAT	3-193
WRITE	3-194
+ or -	3-203

**Section 4. CANDE Control Commands**

?ASDU	4-2
?AT	4-3
Using Host Services	4-4
Using the FTAM Service	4-5
Using FTAM and the CHANGE Command	4-5
Using FTAM and the FILES Command	4-5
Using FTAM and the LFILES Command	4-5
Using FTAM and the REMOVE Command	4-7
Examples	4-7
?AX	4-11
?BREAKPOINT	4-12
?C	4-13
?CHAR	4-14
?% (Comment)	4-16
?CONNECT	4-17
?COUNTS	4-18
?CS	4-19
?CU	4-20
?DENY	4-21

# Contents

---

?DS .....	4-22
?DUMP .....	4-23
ECHO Function .....	4-24
?EDIT .....	4-25
?END .....	4-26
?ENTER .....	4-27
?EOL .....	4-29
?FA .....	4-30
?FR .....	4-31
?GO .....	4-32
?HI .....	4-33
?HN .....	4-35
?ID .....	4-36
?JA .....	4-37
?LIBS .....	4-38
?MCS .....	4-39
?MM .....	4-40
?MSG .....	4-41
?MXA .....	4-42
?NF .....	4-43
?NORESTART .....	4-44
?NOTOK .....	4-45
?NUMBERED .....	4-46
?OF .....	4-47
?OK .....	4-48
?OT .....	4-49
?PURGE .....	4-50
?REPEAT .....	4-51
?REPORT .....	4-52
?RESTART .....	4-53
?RESUME .....	4-54
?RETRIEVE .....	4-55
?RM .....	4-56
?RO .....	4-57
?S .....	4-58
?SC .....	4-58A
?SCHEDULE .....	4-59
?SF .....	4-60
?SHOW .....	4-61
?SI .....	4-62
?SL .....	4-63
?SO or ?RO .....	4-64
?SQ .....	4-65
?SS .....	4-66
?ST .....	4-67
?STARTTIME .....	4-68
?STATUS .....	4-69
?STUP .....	4-71
?TAKE .....	4-72
?TD .....	4-74
?TF .....	4-75
?TI .....	4-76

?TIME .....	4-77
?TO .....	4-78
?UNNUMBERED .....	4-79
?W .....	4-80
?WAIT .....	4-80A
?WD .....	4-81
?WHERE .....	4-82
?WHY and ?Y .....	4-83
?WM .....	4-84
?WRU .....	4-85
?WS .....	4-86
?WT .....	4-87
?Y .....	4-88

**Section 5. Functional Command Groupings**

**Appendix A. Understanding Railroad Diagrams**

What Are Railroad Diagrams? .....	A-1
Constants and Variables .....	A-2
Constraints .....	A-2
Following the Paths of a Railroad Diagram .....	A-5
Railroad Diagram Examples with Sample Input .....	A-6

Glossary .....	1
----------------	---

Bibliography .....	1
--------------------	---

Index .....	1
-------------	---



# Figures

A-1.	Railroad Constraints .....	A-5
------	----------------------------	-----



# Tables

2-1.	Record Formats .....	2-8
3-1.	File Attributes for the ALTER command .....	3-4A
3-2.	File Attributes for the COPY Command .....	3-34
3-3.	File Attributes for the Destination Volume Attribute List .....	3-35
3-4.	FTP Transform Attributes for Copying Files .....	3-37
3-5.	FTAM Transform Attributes for Copying Files .....	3-38
3-6.	Print Attributes and Corresponding Mnemonic Values .....	3-123
3-7.	Print Modifiers and Corresponding Mnemonic Values .....	3-123
3-8.	File-Equatable Files .....	3-185





# Section 1

## General Information

### Pseudostations and COMS Window Dialogs

The Communications Management System (COMS) message control system (MCS) is a Unisys product which, among other features, provides multiple logical connections between CANDE and a single physical station. Each logical connection is treated by CANDE as an individual station and is implemented as a data comm pseudostation. COMS calls each connection a dialog of the CANDE window. Refer to the *A Series Communications Management System (COMS) Operations Guide* for information on logging onto COMS, using windows and dialogs, and COMS commands.

These pseudostations (which are also used for stations transferred from a foreign host across a BNA network) are treated by CANDE almost identically to physical stations. However, because all input to and output from such stations is filtered through another MCS (for example, COMS), there can be differences in behavior. In particular, commands intended for CANDE can instead be intercepted and acted upon by COMS, and output for the station can be stored by COMS, which can alter the usual pattern of flow control.

If any control commands do not yield normal results when entered from a COMS window dialog, the problem can be resolved by using an extra control character (usually a question mark [?]). For example, COMS intercepts a ?WRU command, but entering ??WRU directs the command to CANDE. Sometimes abbreviating the command can be helpful (COMS does not recognize abbreviations). For example, a ?PURGE command will go to COMS, but ?PURG will go to CANDE.

When a dialog of the CANDE window is opened, COMS notifies CANDE of the usercode and if privileged or control status should apply. If the usercode does not require that a chargecode or accesscode be entered, CANDE then automatically logs on the user in a new session. Chargecodes and accesscodes are discussed later in this section. The CANDE *HELLO* command can be used to change to a different usercode.

### User Identification and Logging On

Before you can use most of the CANDE commands, your station must be active and logged on to the system. In order to log on, a valid usercode, and in most cases a password, must be presented to the system. A given usercode can be assigned none, one, or several passwords in the USERDATAFILE. The CANDE *PASSWORD* command can be used to change the list of passwords for a usercode in the USERDATAFILE.

To log on, enter the usercode without the password. The system then requests the password to be entered in a protected field on the screen. If no password is assigned to the usercode, enter a period (.). If no password is assigned to the usercode and the

## General Information

---

usercode is allowed one or more passwords, you can assign a password to that usercode by entering the desired password.

Note that the protected field must be installed in the Network Definition Language II (NDLII) for an installation by having the NDLII recognize that messages with TOG[1] set require a protected field. For installations without the protected field capability, password security can be obtained in the following ways:

- If the terminal is a hardcopy device, enter the password into a blacked-out area.
- For screen devices, password security can be obtained by dimming the screen before entering the password, and then blanking the screen before restoring brightness.

There is an optional method of logging on that can be used on systems in which the SECOPT CLASS option is set to U (unspecified) or S0. Refer to the *A Series Security Administration Guide* for additional information about the CLASS option. To log on, enter the usercode followed by a slash (/) or space and then the password. If no password is assigned to the usercode, enter a period (.) following the usercode.

CANDE checks the usercode/password against the file of authorized users (USERDATAFILE) of the system. If the usercode has password aging and the password has expired, or if the usercode/password is not valid, the user is informed and asked to re-enter the usercode/password. If the usercode/password is valid, CANDE responds with an appropriate message and the user is logged on. If the usercode has password aging and the password is about to expire, CANDE indicates the number of days before the password becomes invalid. In general, CANDE cannot be used unless a successful log on occurs.

If a user fails to provide a valid usercode and password in ten consecutive log-on attempts, the station is made NOT READY and the following message is displayed:

```
STATION CLEARED BECAUSE OF SECURITY VIOLATIONS
```

If the station is on a switched line, the line is disconnected. If the station is not owned by CANDE, it is returned to the controlling MCS.

Although 10 log-on attempts is the system default, the number of log-on attempts can be changed to suit the security needs of the site. You can set the number of log-on attempts by specifying a value in the LOGONATTEMPTS option of the SECOPT system command. The valid values range from 0 (zero) to 15. Values 1 through 15 indicate the number of log-on attempts permitted, and 0 (zero) means that a maximum number of log-on attempts is not enforced.

In addition to a usercode and password, some installations require another accesscode and accesscode password, chargecode, or both as part of the log-on procedure. An accesscode restricts access to certain files and provides an additional layer of security over the usercode/password. A chargecode keeps track of the charges for computer time.

If a chargecode or an accesscode/accesscode password is required for a particular usercode (as specified in the USERDATAFILE), CANDE emits an appropriate message after the user has given the usercode/password.

Accesscode/accesscode password combinations are entered in a manner similar to usercode/password combinations. On systems where the CLASS security option is set to UNSPECIFIED or S0, you can enter the accesscode password directly after the accesscode, separated by a slash (/) or space. On all systems, you can enter the password on a separate line (using a protected field). If you do not have an accesscode password, enter a period (.) instead of the accesscode password.

## General Information

---

If a valid chargecode or accesscode/accesscode password is given, log on is completed. If the chargecode or accesscode/accesscode password is found to be invalid when checked in the USERDATAFILE, an error message is issued. A default chargecode can be assigned to a given usercode in the USERDATAFILE. If the default chargecode is set for the installation, it is automatically invoked at log-on time without any prompt to the user. (For further information about the USERDATAFILE where chargecodes, default chargecodes, usercodes, passwords and accesscodes are set up for an installation, refer to the *Security Administration Guide*.) For information about how to manipulate passwords, chargecodes, accesscodes, and accesscode passwords through CANDE commands, refer to the PASSWORD, CHARGE, ACCESS, and APASSWORD commands in Section 3.

When you use CANDE through a COMS window, COMS can pass enough information to CANDE to allow your station to be automatically logged on. (Refer to "Pseudostations and COMS Window Dialogs" for more information.)

To log on to a different usercode once you are already logged on, use the HELLO command. To log off of a CANDE session, use the BYE command.

## Startup Files and Restart Files

Startup files are ordinary text files that contain CANDE commands. They are identical to DO files, except that for startup files, CANDE invokes the DO command when a user logs on. Logging on in this case includes the initial log on, automatic log on via COMS windows, and invocation of the HELLO command. Startup files are not invoked by commands that cause a new session to be created without a change of usercode, such as the SPLIT, CHARGE, and ACCESSCODE commands.

If the startup file facility is enabled, CANDE searches for a particular startup file when a user logs on. CANDE first searches for a file called <startup name>/<station name>. If this file is not available, CANDE then searches for a file called <startup name>. In both cases, the normal file search rules apply; that is, files are first searched for under the user's usercode, and then under the asterisk (\*) usercode, on the user's primary family. If a file cannot be found on the user's primary family, an alternate family, if in effect, is searched. If CANDE cannot find a file, the search terminates.

Default files can be set up under the asterisk (\*) usercode, including separate defaults for particular terminals and windows. These default files can then be overridden by user files of the same name. The user files can invoke the default files if a DO command is included in the user files.

The startup file is not processed when the work file from the previous session is recovered automatically. If the startup file facility is enabled and an automatic recovery of the work file is performed, CANDE searches for a file called <restart name>/<station name>. If this file is not available, CANDE searches for a file called <restart name>. In either case, the normal file search rules apply as with the startup file search. If CANDE cannot find a file, the search terminates. If the <restart name>/<station name> file or the <restart name> file exists, then the commands in the file are processed before the work file is recovered.

## General Information

---

Like startup files, restart files are text files that contain CANDE commands. Unlike startup files, restart files can perform any conditioning requests that are not part of the recovered work file state. For example, the user's startup file might contain the command to run a mail facility program. The user must process any existing mail and then quit before the system can identify any recovery files. Although an active program does not prevent automatic recovery, it defers the automatic recovery until the program initiated by the startup file is terminated. Using a different restart file or using no restart file allows the recovery action to proceed.

Restart and startup files can also be used indirectly. For example, the restart file might include all commands that condition the session for the user, such as TERM specifications, or ?RO and ?SO commands that are not defined in the USERDATAFILE. The startup file could contain the command DO <restart file>, followed by commands that the user wants performed only upon initial log-on.

### Example

If the <startup name> is CANDE/STARTUP, user FRED is logged on to station STA1, and the USERDATA-defined family substitution statement DISK = PRIM OTHERWISE ALT is in effect, then files are searched for in the following order:

1. (FRED)CANDE/STARTUP/STA1 ON PRIM
2. \*CANDE/STARTUP/STA1 ON PRIM
3. (FRED)CANDE/STARTUP/STA1 ON ALT
4. \*CANDE/STARTUP/STA1 ON ALT
5. (FRED)CANDE/STARTUP ON PRIM
6. \*CANDE/STARTUP ON PRIM
7. (FRED)CANDE/STARTUP ON ALT
8. \*CANDE/STARTUP ON ALT

The search for the restart file is similar to the search for the startup file.

## Accesscodes and Recovery Files

CANDE stores the current accesscode of a CANDE session in the tankfile (a file CANDE maintains for storage) for each user when a work file is created with a MAKE or GET command. To recover a file, the accesscode on the session must be the same as in the tankfile if the accesscode is not null. If an attempt is made to recover a work file with a different accesscode, CANDE sends the following error message:

```
#INCOMPATIBLE ACCESSCODE
```

The recovery file is not recovered or purged. If the recovery file has an accesscode, the accesscode is indicated in the list of recovery files by a special character following the recovery number as follows:

# Accesscode is different from the accesscode of the session.

\* Accesscode is the same as the accesscode of the session.

Refer to "Recovery Files" in this section for more information about recovery files.

## ACCESSCODE Task Attribute

CANDE recognizes the ACCESSCODE task attribute. (Refer to the *A Series Task Attributes Programming Reference Manual* for additional information about task attributes.) Accesscodes are required only for tasks accessing files that are protected by a guard file that uses accesscodes to control access rights. For example, assume a session is running under usercode UA with accesscode/accesscode password AA/PA. The following is entered:

```
RUN (UB)P ON HISPACK; ACCESS AB/PB
```

AB is the accesscode for the task (UB)OBJECT/P. At task start-up time, AB/PB must be a valid accesscode/accesscode password for user UA; if not, the task is not initiated. Access to the program file (UB)OBJECT/P is determined by the security of the program code file; if the security is GUARDED or CONTROLLED, access is determined by the usercode UA and the accesscode AA of the session. Supplying an ACCESSCODE task attribute on a task does not change the accesscode of the session. The ACCESSCODE task attribute cannot be provided as a task attribute to a program being compiled from CANDE. For example, COMPILE; COMPILER ACCESS = A/B is valid; however, COMPILE; ACCESS = A/B is not valid.

## Multiple Commands

More than one CANDE command (control commands excluded) can be entered on a single line by separating the commands with semicolons (;). The commands ADD, BYE, COPY, FIX, HELLO, SEQ, TAPE, WFL, and Single Line Entry or Deletion, however, must appear alone on a line or as the last command of that line. This information is shown in the railroad diagram of each command; a bar (|) termination symbol indicates that multiple entry is legal, and a percent (%) symbol indicates that this command must be the last or only command on that line.

When task-initiating commands (such as RUN, EXECUTE, COMPILE, WRITE, WFL, and START) are entered, modifiers that have the same names as valid CANDE commands (for example, DESTNAME, PRINTDEFAULTS, the *FILE* abbreviation of *FILES*, and FAMILY) and follow these commands on the same line are treated by CANDE as modifiers for the initiated task, not as separate commands. For example, if *FILES* follows a task-initiating command, CANDE interprets it as a separate command. However, if *FILE* follows a task-initiating command, CANDE interprets it as a modifier. Note that CANDE does not look for trailing commands on the same line with commands handled by WFL, such as WFL, COPY, ADD, and PRINT. In such cases, the entire line is processed by WFL instead of CANDE.

## Family Substitution

All CANDE commands that reference files, except for the FILES, LFILES, CHANGE, TITLE, REMOVE, and SECURITY commands, always invoke family substitution. Family substitution is a method for redirecting references to files on a family (such as DISK) in order to avoid entering the actual family name in commands. The following paragraphs describe how the family substitution feature operates. Note that family substitution is effective for CANDE functions only if the target family is DISK. For an explanation of the terminology used in this discussion (family specifications, substitute family, target family, and alternate family), refer to the FAMILY command in Section 3, "CANDE Commands."

If family substitution is desired, and the target family of the family specification is DISK (for example, FAMILY DISK = SUBPK OTHERWISE ALTPK), then whenever DISK is specified as the family name in a command (for example, LIST MYFILE ON DISK), family substitution takes effect. This means that in looking for the file to list, the system searches the substitute family pack and, possibly, the alternate family pack (SUBPK and ALTPK in this example).

For commands involving input from an existing file (such as the LIST or RUN commands), the alternate family, when present, is also searched if the attempt to locate the file on the substitute family should fail (for example, file MYFILE ON ALTPK will be used if file MYFILE ON SUBPK cannot be found).

If file MYFILE ON DISK is requested, and MYFILE is actually located on DISK and nowhere else, the file will be inaccessible to commands other than the commands listed previously as exceptions, unless the aforementioned FAMILY statement is changed to include DISK (for example, FAMILY DISK = SUBPK OTHERWISE DISK or FAMILY DISK = DISK ONLY), or the FAMILY specifications are deleted entirely.

When family substitution is in effect, new files that request the target family are created on the substitute family.

In the case of the commands listed previously as exceptions, family substitution is ignored if a family name is specified. That is, if the family substitution in effect is FAMILY DISK = SUBPK OTHERWISE ALTPK, then entering the command FILES is equivalent to entering FILES ON SUBPK; however, specifying FILES ON DISK yields files on DISK and not on SUBPK.

## Break Condition

Although ?BRK is not a CANDE command, standard Unisys data comm software issues a break condition when a user enters ?BRK or presses the break key. When a break condition occurs, CANDE immediately notifies any output-capable open remote files assigned to the user's station, and terminates any listing or similar activity that can generate output to a station. For example, ?BRK terminates the FIND, MATCH, and LIST commands.

If a program has no instructions for recognizing a break condition, the program is discontinued with the *break on output* I/O error. If a program has instructions for recognizing a break condition, the action taken depends on the provisions in the



program. For additional information, refer to the *A Series I/O Subsystem Programming Guide* and the appropriate language reference manual.

## Response to CANDE Commands

Except for single-line entry and certain control commands, all commands sent to CANDE result in the display of a number sign (#) prompt, either alone or accompanied by a message. Progress in processing a command may be indicated by a number sign (#) with a response (for example, "#UPDATING"). If the command has not been successfully completed, the number sign is followed by an error message. Successful completion of a CANDE command results in the display of the number sign (#) alone or with messages related to the command's results.

## Saved Text Queue Manipulation

Each user has a queue of saved text that CANDE maintains for reference. The most recent saved text can be edited and processed as the next line of input. The commands provided for listing and editing saved text are ?SHOW, ?EDIT, ?REPEAT, and ?RETRIEVE.

Each normal line of input (a line not beginning with the control character defined for the user's station) becomes the current saved text. Each entry in the queue is then moved up one level. If the saved text queue contains the maximum number of entries, the oldest entry is discarded. The maximum number of entries that can be saved in the saved text queue can be altered by the CANDE control command ?DEPTH. You must be a CANDE control station to use the ?DEPTH command to change the saved text queue entry limit. The maximum number range from 0 to 20, inclusive. Refer to the *A Series CANDE Configuration Reference Manual* for more information about the ?DEPTH command.

If a line of input contains more than one CANDE command, each command is entered separately in the saved text queue. If a command does not complete successfully, the most recent saved text entry contains the failed command. Commands that follow the failed command are either part of this entry or are the first entry in the input queue (depending on the nature of the failure).

Throughout this manual, the use of saved text without a queue entry number refers to the most recent entry.

## Input Queue Manipulation

CANDE can execute an unlimited number of queued entries. Queued input is in one of three states: normal, pending, or waiting.

## General Information

---

- The normal state occurs when CANDE is busy executing a command for a station and another command is entered before the first is completed. The message "#QUEUED" is issued to the user, indicating that the new command has been placed in the input queue. If the currently executing command completes normally, then the first queued command becomes the currently executing command, and so on. If an error occurs on the currently executing command and queued input exists, the queued input is then either pending or waiting. (Refer to the QWAIT option below.)
- The pending state occurs if a "#QUEUED INPUT PENDING" message is given; if a normal line of input is entered after the message, then the queued input is discarded and the session continues with that line of input. If the queued input should not be discarded, then one of the queue manipulation commands can be used and the input queue can be set to the waiting state.
- In the waiting state, any normal line of input is queued at the end of the queue and the message "#QUEUED" is given. A "#QUEUED INPUT WAITING" message is then displayed. The state of the queued input remains waiting until a ?GO, ?PURGE, or ?REPEAT command is issued.

While the state of the queued input is waiting or pending, queued input entries can be entered, removed, or edited. Insertion of an entry specifies that CANDE is to wait for intervention before performing that entry or that the session is resumed with the first entry in the queue or with the saved text.

Certain queue manipulation commands are limited to operation on the visible portion of the user's queue. Only the first 20 entries of the user's queue are visible.

QWAIT is a user option that is set or reset with the SO (or ?SO) and RO (or ?RO) commands, respectively. If QWAIT is set and an error occurs, the state of the user's queued input is set to waiting; otherwise, it is set to pending.

The commands provided for manipulation of queued entries are ?GO, ?WAIT, ?PURGE, ?TAKE, and ?ENTER.

## Attributes of CANDE Files

CANDE can use files with 1-character records.

Files created through CANDE have the following attributes by default:

```
KIND = DISK
UNITS = WORDS (CHARACTERS for CDATA and CSEQDATA)
MAXRECSIZE = 14, 15, or 80 (refer to Table 2--1)
MINRECSIZE = 0
BLOCKSIZE = 420 (2160 for CDATA and CSEQDATA)
AREAS = 15
AREASIZE = 504 (1134 for CDATA and CSEQDATA)
SAVEFACTOR = 30
SECURITYUSE = IO (refer to the MAKE command)
SECURITYTYPE = PRIVATE (refer to the MAKE command)
```

## Limitations of CANDE Files

Files processed by CANDE must meet the following requirements and structural limitations:

- The **BLOCKSIZE** attribute must be greater than 32 characters (if **UNITS=CHARACTERS**), or greater than or equal to 6 words (if **UNITS=WORDS**). The **BLOCKSIZE** attribute must also be less than 3,060 characters (if **UNITS=CHARACTERS**), or less than 510 words (if **UNITS=WORDS**).
- The **BLOCKSTRUCTURE** attribute must be **FIXED**.
- The **EXTMODE** attribute must be **EBCDIC** or **ASCII**.
- The **FILEORGANIZATION** attribute must be **NOTRESTRICTED**.
- The **INTMODE** attribute must be **EBCDIC** or **ASCII**.
- The **KIND** attribute must be **DISK** or **PACK**.
- The **MAXRECSIZE** attribute must be less than 255 characters (if **UNITS=CHARACTERS**), or less than or equal to 42 words (if **UNITS=WORDS**).
- With the exception of the **WRITE** command, CANDE cannot process sequenced files containing more than 1,048,575 records, or data files containing more than 999,999 records.
- The **WRITE** command does not recognize a file name that exceeds 11 nodes (excluding the usercode).

### Work Files

CANDE can access any file in the user's library for various purposes, but changes can be effected only on the work file. A new work file can be made using the MAKE command, or an existing file can be used as the work file using the GET command. All editing commands apply to the work file. Single-line entries, as well as FIX commands and page mode input, are not applied to the work file immediately but are tanked until the next UPDATE.

### Recovery Files

The work file currently being updated resides in a TEXT file separate from the compiled object work file that exists as a CODE file. All other information about an active work file, including any changes since the last update, is kept in the tankfile. A recovery file is created by transcribing the information in the tankfile that pertains to the station whose session was aborted. This transcription is performed immediately, or when CANDE is next initiated.

Recovery information is contained in three files:

- A RECOVERY file contains any work file changes since the last update, as well as the title and other attributes of that work file. This file has the title CANDE/RECV/<recovery number> .
- A TEXT file is created if the work file has been updated but not yet saved. This file has the title CANDE/TEXT <recovery number > .
- A CODE file is generated if the work file has been compiled but not yet saved. This file has the title of CANDE/CODE <recovery number > .

A TEXT or CODE file is generated at update or compilation time and is written onto the work file family. The recovery file is produced when or after the session is aborted and can be written on one of two families. If possible, the recovery file is written on the default work file family, as defined by the work file family specifications established by the user at log-on time. If that family is not available, the recovery file is written on the family containing the code file for the CANDE MCS.

The recovery number consists of the logical station number (LSN) (in decimal) followed by a digit to distinguish among multiple recovery files from the same station. The recovery number for a session is determined at the beginning of the session by using the system; this number is suffixed to the TEXT and CODE files created by updating and compiling the work file. If a recovery file must be created, the same number is suffixed to the RECV file.

This scheme imposes a limit of 10 recovery files from the same station and a total of 25 recovery files under any one usercode. (The second limit is an arbitrary define, MAXRECFILES, that the installation can modify by compiling CANDE; its upper bound is 149.) If 25 or more recovery files exist, only the first 25 are listed at log-on time or by the RECOVER command, and any attempt to get or make another work file is rejected with the message:

```
#RECOVER OR DISCARD A WORKFILE.
```

For example, if 10 recovery files exist for LSN 23, any GET or MAKE or an attempt to recover a file created from another station is rejected with a message such as the following:

```
#RECOVER OR DISCARD A WORKFILE IN THE RANGE 230-239
```

If one or more recovery files exist and the AUTORECOVER option of the USERDATAFILE file attribute is set to TRUE, then recovery files are not displayed when a user logs on. The AUTORECOVER option causes an automatic attempt to recover recovery files created under the user's usercode and station.

A high level of consistency checking is applied to recovery files to screen out files harmful to CANDE. If a CANDE fault or error occurs in a work file editing or output operation (a CANDE worker), the following actions are taken to invoke the consistency checking of CANDE work file recovery:

1. The tankfile data is saved in a recovery file, as though the station had disconnected.
2. CANDE displays the message "#AUTORECOVERY INITIATED".
3. The action of a RECOVER command is taken using the appropriate recovery file.

If the consistency checking fails, normal invalid recovery file action follows. The contents of the RECV file are listed in the CANDE tankfile to permit diagnosis of the failure. The file is then purged. If a TEXT work file exists (that is, if the file was updated since GET or MAKE), then that file is recovered. The results of such recovery follow:

```
#WORKFILE IS NOT NAMED:  ALGOL, 347 RECORDS
```

```
#INVALID RECOVERY FILE; NAME AND ANY CHANGES WERE LOST
```

If no text existed or if the recovery failed, the message is

```
#INVALID RECOVERY FILE
```

The RECOVER command displays and recovers recovery files in up to three places in the following order of precedence:

- The USERDATAFILE default work file family
- The CANDE MCS family
- The current work file family

The DISCARD command removes RECV, TEXT, and CODE files with the specified number or numbers from all three of these families.

A display listing recovery files is grouped according to the family containing the recovery files. If a recovery file pertains to a work file on a different family, the phrase *ON <family name>* appears in the display.

## General Information

---

Recovering a work file sets the session specifications LANGUAGE, CONVENTION, and PRINTDEFAULTS to those in effect when the work file was saved. New session specifications are displayed if different from those in effect before recovery.

If the file part of a recovered work file is not present, the recovery action is aborted with an appropriate message. However, since the system does not purge the recovery file, you may attempt a recovery by making the missing file present with a GET command.

## Remote Files

Remote files are of a special nature in that an object program can treat a remote station as a file by setting the KIND file attribute to REMOTE. (Refer to the *A Series File Attributes Programming Reference Manual* for additional information about setting file attributes.)

More than one open file can be assigned simultaneously to a station. However, more than one input or input/output (I/O) file cannot be open at the same station at the same time.

When a remote file is opened for input or input and output, all input received from the station is considered input to the file and not a command to CANDE. The only exception occurs when a line is preceded by the control character of the station. In that case, the input is to be a control command and is sent to CANDE for processing.

You can direct output to another CANDE station by specifying the LSN of the station through the STATION <task equation list> construct (see the discussion of the <task equation list> construct in Section 2, "Basic Constructs"). For example, the following command directs the remote output to LSN 15:

```
EXECUTE;STATION=15
```

You can also assign remote files to other CANDE stations by using the TITLE attribute and setting the STATION <task equation list> to zero. For example, the following command directs file R to the station name TTY3:

```
EXECUTE;STATION=0;FILE R(TITLE=TTY3)
```

## Page Mode Operations

Page mode allows a full page (that is, screenful) of text to be entered and edited at one time and enables the user to move back and forth through the work file easily while executing other CANDE commands.

## Requirements for Page Mode

Page mode is available only for users of screen terminals that can receive, display, and transmit at least three lines of data at a time. The number of lines a terminal can receive and display is calculated from the PAGESIZE, LINEWIDTH, and MAXOUTPUT of the

terminal; the default values for these terminal attributes are defined in NDII, and they can be checked and adjusted by the CANDE *TERMINAL* command. The terminal's display capacity equals the smaller of either the PAGESIZE (number of lines received on the terminal screen at a time) or MAXOUTPUT (the number of characters a screen can display and hold in display terminal buffer memory) divided by LINEWIDTH (characters per line):

```
PAGESIZE = 24 lines
MAXOUTPUT = 1000 characters
LINEWIDTH = 80 characters
```

For example, given the previously listed limits, the terminal's display capacity (the size of a page) is 12 lines, which is the smaller of PAGESIZE (24 lines) and MAXOUTPUT divided by LINEWIDTH (1000/80 = 12 lines).

MAXINPUT is the number of characters a terminal is capable of transmitting at one time. MAXINPUT is defined in NDII for a particular installation and must coincide with the capability of the terminal. The TERMINAL command is used to list the MAXINPUT (refer to the TERMINAL command in this manual). If the MAXINPUT is less than the MAXOUTPUT, the terminal cannot transmit the displayed page to CANDE and the full page mode editing capabilities cannot be used. For the best results in using page mode, MAXOUTPUT and MAXINPUT should be equal.

CANDE displays PAGESIZE\*TERMWIDTH characters for each page, as well as PAGESIZE\*2 characters (Carriage Return and Line Feed), if WRAPAROUND is FALSE. A page can be distorted if a particular terminal's data comm buffer size is not large enough to accommodate the number of characters CANDE sends. (Refer to your terminal's manual for an explanation of how to set the data comm buffer size.)

## Initiating Page Mode and Moving through the Work File

The commands that initiate page mode and allow movement through the work file are briefly described in the following list. For more complete descriptions with examples, see Section 3, "CANDE Commands." Any of the five commands given below, except + and -, will initiate page mode and display the first page of records in the work file with the token NEXT + in the upper left-hand corner on an unnumbered line. The exceptions are + and -, which always shift the indicated increment before displaying a page of records.

After page mode is initiated, the page-invoking commands do the following:

Command	Definition
PAGE	Displays a page starting with a specified sequence number.
NEXT +/-	Displays a page after shifting from the currently displayed page forward (+) or backward (-) a specified number of pages.
SAME	Refreshes the currently displayed page.
+, -	Displays a page after shifting from the currently displayed page forward (+) or backward (-) a specified number of records.

*continued*

## General Information

---

*continued*

Command	Definition
SEQ	Displays a page with a column of sequence numbers, determined by the specified base and increment, at the leftmost locations on the screen. When SEQ is transmitted after a text already exists, the previously sequenced records are numerically inserted along with the new sequence numbers. As each page is transmitted, the next page of sequence numbers is displayed. If a CANDE command other than SEQ is transmitted, sequencing automatically stops. Sequencing resumes where it left off when a SEQ NEXT is transmitted.

### The Column Indicator

Once page mode has been initiated, a column indicator appears on the top line of the screen that specifies actual text columns for a record in the file. The indicator is determined by the file type. The file type CSEQ has a blank column 6. In this case, a space for column 6 does not appear on the screen. When nonblank data is transmitted in columns past the end of the indicator but within the text field, the record is changed. If the nonblank data is not within the text field, an error message is displayed. If the sequence number field is shorter than five characters for a file type, the word NEXT + overflows into the column indicator.

### Entering Text and Commands in Page Mode

A whole page of text is transmitted to CANDE when the cursor is placed at home position (the upper left-hand corner of the screen) or at the end of the last line of text on the page and the XMIT (transmit) key is pressed. NEXT must always be the first token on the screen for text to be transmitted in page mode. When a page is transmitted that does not contain any CANDE commands other than VOID, a new page is displayed that begins with the last line transmitted. When the XMIT key is pressed, CANDE receives everything between the home position and the cursor if the cursor is not at home position.

The setting of WRAPAROUND, which can be checked and modified by the TERMINAL command, affects how lines of text are ended. If WRAPAROUND is set to FALSE, each line of text ends with a carriage return and linefeed so the text can be displayed on the screen properly. If WRAPAROUND is set to TRUE, the terminal automatically does a linefeed and carriage return.

Records must be transmitted in sequential order or a sequence error is flagged and record processing stops. The following message is given when the end of the file or the end of the sequence range requested by a page-invoking command has been reached:

```
#DISPLAY COMPLETE
```

Records can be inserted between the records given by CANDE. The sequence field can be left blank if a record prior to these inserted lines has a sequence number. CANDE calculates the sequence numbers for these inserted lines, and the largest sequence increment allowing all records to be numbered is assigned. The increments considered



are 100, 50, 20, 10, 5, 2, and 1. If the records do not fit when the minimum increment of 1 is used, a message is displayed. Under certain conditions, an error message might be displayed although there is sufficient room for the inserted lines. Refer to the examples later in this section for an explanation of how this can occur.

Resequencing may be done on records with existent sequence numbers so that all records can be assigned a sequence number. Sequence numbers can be entered manually if the numbers are lined up with the existing sequence numbers; the leading zeros can be replaced by blanks if desired. For example, if *b* signifies a blank space, 00000100 can be typed as *bbbb100*. Unnumbered blank lines transmitted at the bottom of the page are ignored. However, if blank lines are desired at the bottom of the page, a sequence number or a command must be typed on the last of the blank lines.

Any valid CANDE command can begin in the first column of any line within the currently displayed page. However, transmitting any commands, except the five page-invoking commands and the VOID command, interrupts page mode. CANDE processes page mode input record by record until a nonpage mode command other than VOID is found. At this point CANDE processes the remaining text on the screen as a command or string of commands separated by semicolons or carriage returns. If a CANDE command followed by additional page mode text is transmitted, an error message is given. The cursor should be located immediately after the command when it is transmitted.

To return to page mode after processing commands other than VOID or a page mode command, one of the page-invoking commands must be used. CANDE keeps track of the most recently displayed page of text as a location in the work file. When page mode is temporarily interrupted to process a nonpage-invoking command, the page-invoking commands can be used to return to page mode at that displayed page (SAME) or near it (NEXT, PAGE, + or -). CANDE maintains the current (or most recently) displayed page after the execution of all other CANDE commands, with the exception of one group. This group consists of the following commands, other than SAVE, that finalize the state of the work file: REMOVE, GET, MAKE, RECOVER, MCS, BYE, and HELLO.

The asterisk (\*) form of the FIX command and the at sign (@) form of the MARGIN command cannot be used within numbered lines while in page mode. (This differs from single-line sequencing mode.)

Page mode input is processed only when the next UPDATE, LIST, or any page-invoking command is entered. Because of this, an error message may not be displayed at the time page mode input is transmitted.

### Examples

In the following example (an ALGOL file), the XMIT key should be pressed when the cursor is in the last column of line 2400. Line 200 is deleted and a new page that begins with the sequence number 2400 is displayed.

## General Information

---

```
NEXT+ .....1....*..    ...    ..*....6....*....7..
0000100 ABC
V000200 DEF <This line will be deleted.>
0000300 GHI
.....
.....
.....
00002400 JKL
```

In the following example, the FIND command is executed after lines 100-300 are processed, if the cursor follows the FIND command when XMIT is pressed:

```
NEXT+ .....1....*..    ...    ..*....6....*....7..
0000100 ABC
V000200 DEF <This line will be deleted.>
0000300 GHI
FIND LIT /ABC/ :TEXT
```

The delay between transmitting a command and receiving an error message about that command is demonstrated in the following example:

```
NEXT+ .....1....*..    ...    ..*....6....*....7..
00001000 ABC
00002000 DEF
00003000 GHI
          JKL
00003001 MNO
DELETE 4000-END
```

Upon the next UPDATE, LIST, or page-invoking command, the following error message is given:

```
#CANNOT SEQUENCE WITHIN BOUNDS AT LINE 5
```

Notice that LINE 5 refers to the fifth line transmitted, the line with NEXT being the first line transmitted.

The following example demonstrates how using a non-page-mode command (DELETE, in this example) while in page mode, and then entering unsequenced lines before using a page-invoking command to return to page mode, can cause CANDE to calculate sequence numbers incorrectly (the file type shown is ALGOL).

The PAGE command, entered on page 1 of the terminal, returns the following screen:

```
NEXT+ .....1....*....2....*    ...    ....6....*....7..
00000005 ABC
00000010 DEF
00000012 GHI
00000014 JKL
00000025 MNO
```

After moving to page 2 of the terminal, the following command is entered:

```
DELETE 12
```

Then, moving back to page 1 of the terminal, the following screen is transmitted when you attempt to enter three unsequenced lines between lines 10 and 14, without first entering a page-invoking command (PAGE or SAME) to return to page mode:

```
NEXT+   ....*....1....*....2....*   ...   ....6....*....7..
00000005 ABC
00000010 DEF
        XXX
        YYY
        ZZZ
<XMIT from here>
```

CANDE returns the error message:

```
#CANNOT SEQUENCE WITHIN BOUNDS AT LINE 4,
LAST VALID SEQUENCE=10
```

Even though line 12 was deleted, since no page-invoking command was subsequently entered, the page mode tables were not updated with the deletion. Therefore, CANDE attempted to place the new unsequenced lines between line 10 and the next known line, which it had saved internally as line 12. The sequence error message was generated because CANDE could not fit three unsequenced lines between lines 10 and 12.

**Note:** *If a command is included in the transmission (after the page-mode input), there could be a delay between transmitting the screen and receiving an error message. This is illustrated in a previous example.*

If PAGE or SAME had been entered upon returning to page 1 of the terminal, the page mode tables would have been correctly updated with the deletion, and there would have been sufficient room for the new lines.

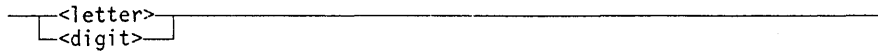


# Section 2

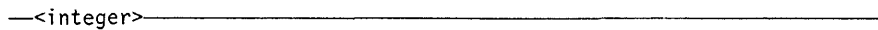
## Basic Constructs

A number of basic constructs are common to CANDE commands. These constructs are defined in this section.

### <alphanumeric character>



### <base>



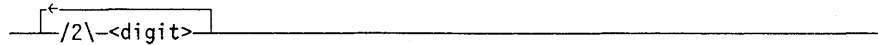
### <character>

Any < alphanumeric character >, any < special character >, or a blank.

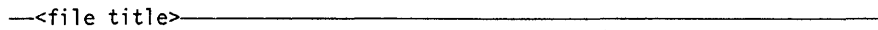
### <code visibility mnemonic>



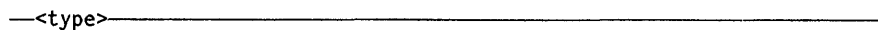
### <column>



### <compiler name>



### <compiler type>



### <control char mnemonic>

NUL SOH STX ETX HT DEL VT FF CR SO SI DLE DC1 DC2 DC3 NL  
BS CAN EM FS GS RS US LF ETB ESC ENQ ACK BEL SYN EOT NAK  
SUB SP

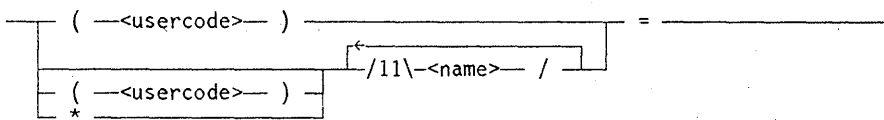
### <digit>

Any one of the decimal digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

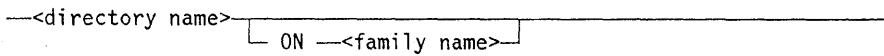
## Basic Constructs

---

### <directory name>



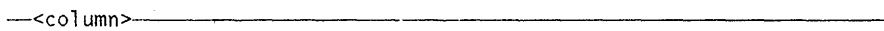
### <directory title>



### <dlm>

The <dlm> character is a delimiter; it can be any special character except the comma (,), colon (:), or at sign (@).

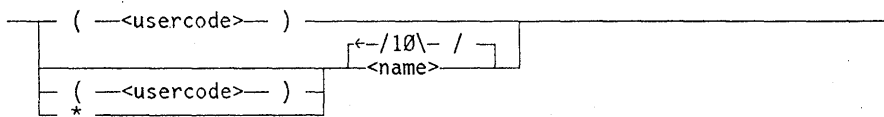
### <end column>



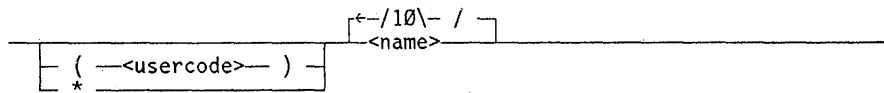
### <family name>



### <file directory>

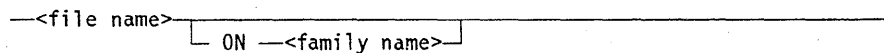


### <file name>



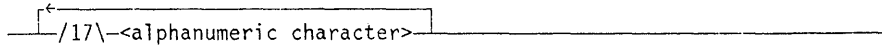
The WRITE command does not recognize a file name that exceeds 11 nodes (excluding the usercode).

### <file title>



On certain commands, if <file name> is a valid <sequence number> value, then it must be enclosed in quotation marks to avoid confusion with a real <sequence number> value. For example, in order to list the file 100-200, you would enter *LIST "100-200"*, instead of *LIST 100-200*. To list lines 100 through 200 of the file 100-200, you would enter *LIST 100-200 "100-200"*. The commands for which this constraint is valid are DO, EXCLUDE, FIND, INSERT, LIST, MATCH, MERGE, RANGE, RMERGE, and WRITE.

**<hostname>**



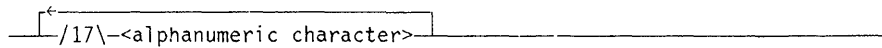
**<hyphen>**

The single hyphen (-).

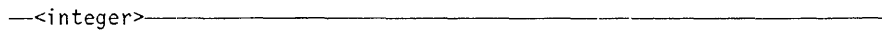
**<ID char>**

Any EBCDIC character for which the hexadecimal code is greater than or equal to 4"40" and which is not a quotation mark (").

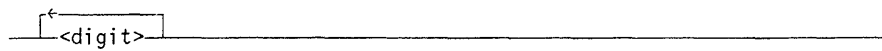
**<identifier>**



**<inc>**



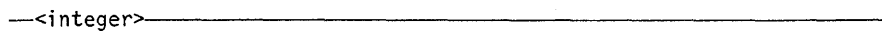
**<integer>**



**<letter>**

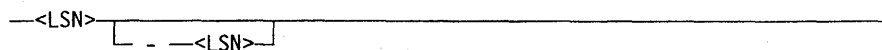
Any one of the alphabetic characters from A through Z, inclusive, in uppercase or lowercase.

**<LSN>**



The logical station number (LSN) is a unique integer assigned by NDLII to each station defined for a network. The most efficient method of station designation is by <LSN>.

**<LSN range>**



The <LSN range> construct refers to a group of one or more LSNs. If a range is defined, the lower LSN must precede the higher LSN.

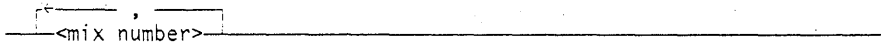
**<mix number>**



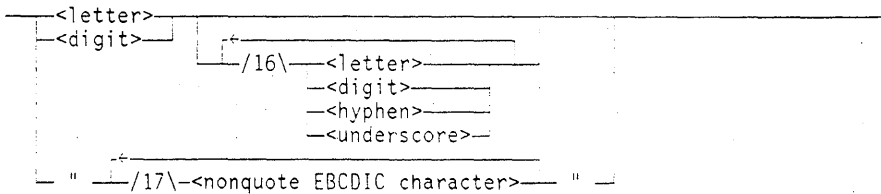
## Basic Constructs

---

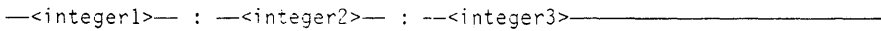
<mix number list>



<name>



<NLS>



The network line station or <NLS> construct identifies a station by the network support processor (NSP) number, line number, and relative station number within the line, as specified in the Network Definition Language II (NDLII) definition for the line to which the station is assigned.

Integer1, integer2, and integer3 represent the NSP, line, and station numbers, respectively. All three must be specified to identify one station on a multidrop line. To determine the <NLS> value for a given station, the following information is required:

- The relative NSP number
- The line number, which is computed by multiplying the cluster or relative Line Support Processor (LSP) number by 16 and then adding the adaptor number
- The station numbers on the line, which are numbered 0 through n-1, where n is the number of stations assigned to the line

When CANDE is initialized via an NSP, users on the NSP receive a message that CANDE is available.

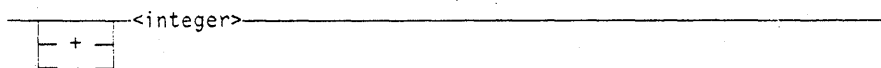
<nonquote EBCDIC character>

Any EBCDIC character for which the hexadecimal code is greater than or equal to 4"40" and that is not the EBCDIC character quotation mark (").

<nonsingle quote EBCDIC character>

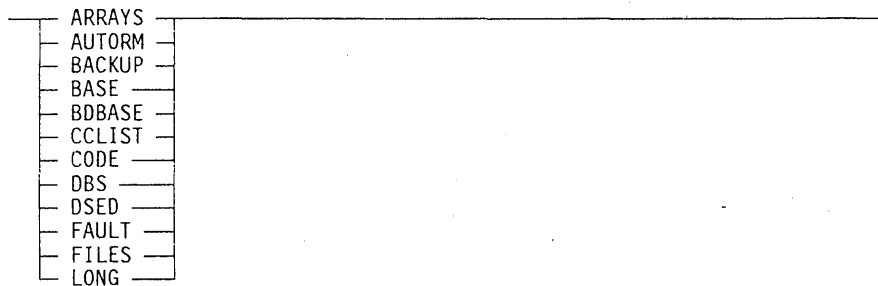
Any EBCDIC character for which the hexadecimal code is greater than or equal to 4"40" and that is not the EBCDIC single quotation mark (').

<number>

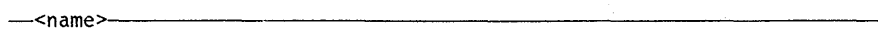




<option list>



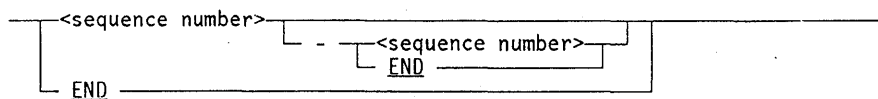
<password>



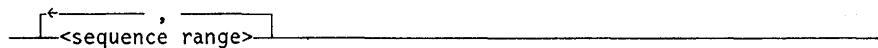
<sequence number>

An integer that represents a value for the sequence field of a record. The maximum number of digits that might comprise a sequence number is determined by the FILEKIND value of the work file. Refer to Table 2-1 later in this section.

<sequence range>



<sequence range list>



<special character>

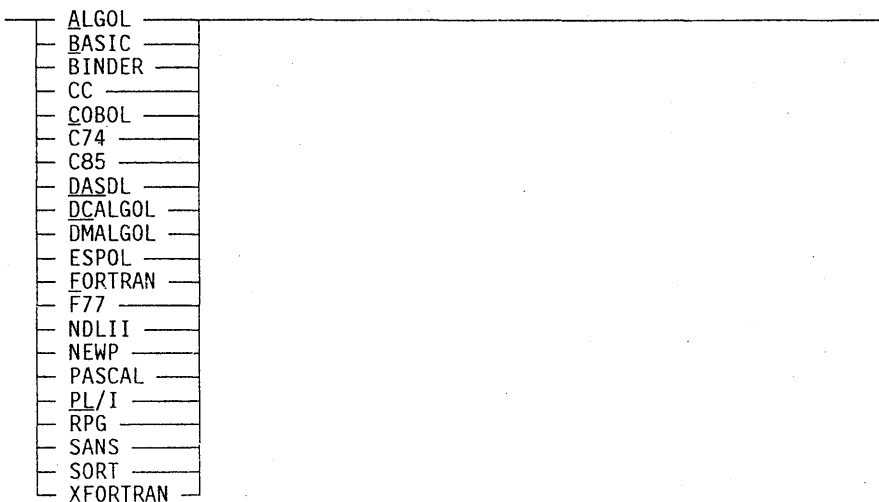
Any one of the following characters:

@ at	( left parenthesis	! exclamation point
# number	) right parenthesis	? question mark
\$ dollar	[ left bracket	' apostrophe (single quote)
% percent	] right bracket	" quotation mark
& ampersand	{ left brace	+ plus sign
* asterisk	} right brace	- minus sign (hyphen)
= equal	< less than	split bar
, comma	> greater than	~ tilde
; semicolon	/ slash	^ circumflex (carat)
: colon	\ backslash	` grave accent
. period	_ underscore	■ DEL (rubout)

## Basic Constructs

---

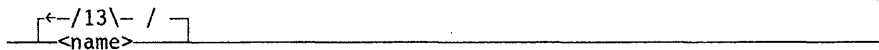
### <standard compiler>



### <start column>

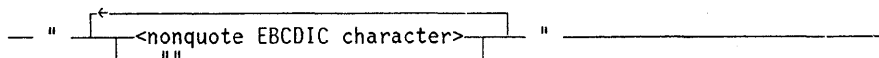


### <station name>

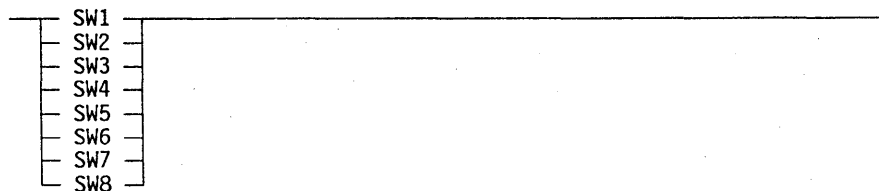


The station name is a unique identifier chosen by the installation for each station that is a member of the network. Station names in NDLII follow the same syntactic conventions as file titles within the system, because any station may be assigned to a REMOTE file.

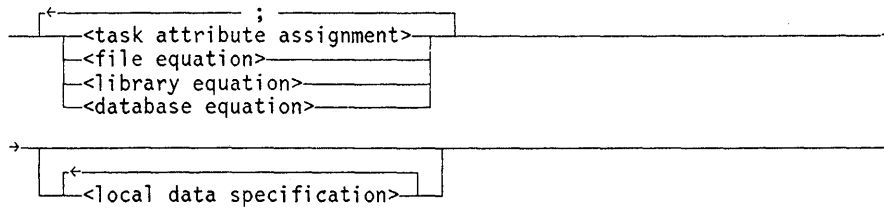
### <string>



### <switch>



<task equation list>



CANDE can process task equations used in WFL statements. Refer to the *A Series Work Flow Language (WFL) Programming Reference Manual* for the descriptions of the task attribute assignment, file equation, library equation, database equation, and local data specification.

<type>

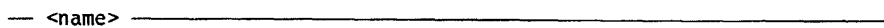
ALGOL
BASIC
BINDER
CC
CDATA
COBOL
CSEQ
C74
C85
DASDL
DATA
DCALGOL
DMALGOL
ESPOL
FORTRAN
F77
GUARD
JOB
NDLI
NEWP
NLS
PASCAL
PLI
RPG
SANS
SEQ
SORT
TEXT
XFORTRAN

The format of the records for each file type is described in Table 2-1.

<underscore>

The underscore character (\_).

<usercode>



## Basic Constructs

<visibility mnemonic>

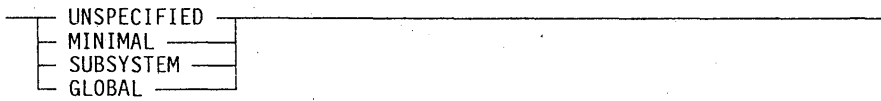


Table 2-1 summarizes the record formats used for each CANDE type.

**Table 2-1. Record Formats**

Type	Text Field	Sequence Field	ID Field	Default Record Length	Minimum Record Length	Compiler Type
ALGOL	1-72	73-80	81-90	15 WD	80 CH	YES
BASIC	5-72	1-4	73-80	14 WD	72 CH	YES
BINDER	1-72	73-80	NA	14 WD	80 CH	YES
CC	1-72	73-80	81-90	15 WD	80 CH	YES
CDATA	1-80†	NA	NA‡	80 CH	NA	NO
COBOL	7-72	1-6	73-80	14 WD	72 CH	YES
CSEQ	7-80†	1-5	NA	80 CH	NA	NO
C74	7-72	1-6	73-80	14 WD	72 CH	YES
C85	7-72	1-6	73-80	14 WD	72 CH	YES
DASDL	1-72	73-80	81-90	15 WD	80 CH	YES
DATA	1-80§	NA	NA‡	14 WD	80 CH	NO
DCALGOL	1-72	73-80	81-90	15 WD	80 CH	YES
DMALGOL	1-72	73-80	81-90	15 WD	80 CH	YES
ESPOL	1-72	73-80	81-88	15 WD	80 CH	YES
FORTRAN	1-72	73-80	NA	14 WD	80 CH	YES
F77	1-72	73-80	81-90	15 WD	80 CH	YES

† CDATA and CSEQ files have an arbitrary length; the text field extends through the end of the record.

continued

‡ For files of type DATA, CDATA, and SCHEDULE, CANDE assigns pseudo-sequence numbers for its own internal use. CANDE computes these numbers by multiplying the relative record number by 100. Each time the file is updated, these pseudo-sequence numbers are recalculated.

§ A DATA file may have an arbitrary length. The text field extends through the end of the record with one exception: by convention, 14-word EBCDIC or ASCII records are assumed to have text only through column 80.

¶ A file of type GUARD is the output file of GUARDFILE and cannot be generated through CANDE.

Table 2-1. Record Formats (cont.)

Type	Text Field	Sequence Field	ID Field	Default Record Length	Minimum Record Length	Compiler Type
GUARD†	1-72	73-80	NA	14 WD	80 CH	NO
JOB	1-80	83-90	NA	15 WD	90 CH	NO
NDLII	1-72	73-80	81-90	15 WD	80 CH	YES
NEWP	1-72	73-80	81-90	15 WD	80 CH	YES
PASCAL	1-72	73-80	81-90	15 WD	80 CH	YES
PLI	1-72	73-80	81-90	15 WD	80 CH	YES
RPG	6-80	1-5	81-90	15 WD	80 CH	YES
SANS	1-72	73-80	81-90	15 WD	80 CH	YES
SCHEDULE	1-80 <sup>††</sup>	NA	NA <sup>‡‡</sup>	80 CH	NA	NO
SEQ	1-72	73-80	81-90 <sup>‡‡</sup>	14 WD	80 CH	NO
SORT	1-72	73-80	81-90	15 WD	80 CH	YES
TEXT	1-72	73-80	81-90	15 WD	80 CH	NO
XFORTRAN	1-72	73-80	NA	14 WD	80 CH	YES

<sup>††</sup> A SCHEDULE file may be of any type recognized by CANDE. (No type SCHEDULE exists.) The output for a SCHEDULE file is type SCHED. Any CANDE command treats a SCHED file as type CDATE.

<sup>‡‡</sup> SEQ files are created with 14 words and no ID field. If CANDE encounters a SEQ file with 15 or more words (90 or more characters), it treats columns 81 through 90 as an ID field. (For example, you can generate such a file by making a file type ALGOL and changing its type to SEQ.)



## Section 3

# CANDE Commands

This section contains the CANDE commands for task initiation, work file creation, user library manipulation, text editing, and so forth. The commands are presented in alphabetical order along with the railroad syntax structure, explanation, and simple examples.

Commands for which the syntax diagram terminates with a percent sign (%) must appear alone on the input record to CANDE or as the last command of that input record. Commands for which the syntax terminates with a vertical bar (|) may appear in sequence on the input record, each command separated from the following command by a semicolon (;).

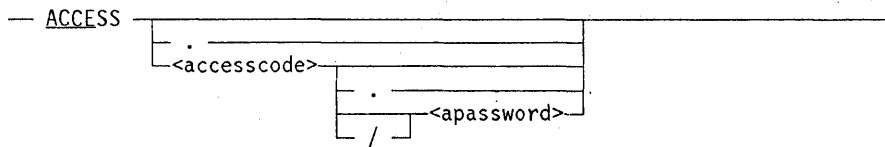
CANDE commands can be abbreviated for convenience. CANDE checks only the first four characters of a command; however, some frequently used commands have shorter abbreviations. For example, the MAKE command can be abbreviated to M, and the SAVE command can be abbreviated to SA. Acceptable forms of the MAKE command are M, MA, MAK, MAKE, MAKEIT, MAKEFILE, and MAKEWORKFILE.

Some CANDE commands directly invoke Work Flow Language (WFL) statements that have the same name, such as ADD, COPY, PRINT, and START. It is not necessary to precede these commands with *WFL*. However, several other CANDE commands that have the same name as WFL statements do not invoke WFL (and in some cases have different syntax or function than the identically-named WFL statement). These CANDE commands are ACCESS, BIND, CHANGE, COMPILE, DO, LOG, PASSWORD, REMOVE, RUN, SECURITY, and STOP. To invoke the WFL statements with these names, you must precede them with the CANDE *WFL* command, or include them in a job file and initiate the job with a START command.

Section 5, "Functional Command Groupings," provides a quick reference to related commands.

## ACCESS

### Syntax



<accesscode>

<name>

<apassword>

<password>

### Explanation

The ACCESS command is used to interrogate or change the accesscode of a session.

When ACCESS is entered, the current accesscode for the session is returned (not including the accesscode password).

The ACCESS . form assigns a null accesscode to the session. If the ACCESSCODENEDED attribute is set in the USERDATAFILE for the usercode, the request is denied.

The ACCESS <accesscode>/<apassword> form assigns or changes the session's accesscode and corresponding accesscode password after validation in the USERDATAFILE. The ACCESS command does not alter the USERDATAFILE. If the <apassword> is not supplied, it is requested.

The ACCESS <accesscode> . form is used when no accesscode password exists for a particular accesscode.

On InfoGuard systems with the SECOPT CLASS option set to S1 or higher, the accesscode password must be entered on a separate line, usually via a protected field.

Changes of accesscodes to a session are logged.

For further information on accesscodes, refer to "User Identification and Logging On" in Section 1, "General Information."



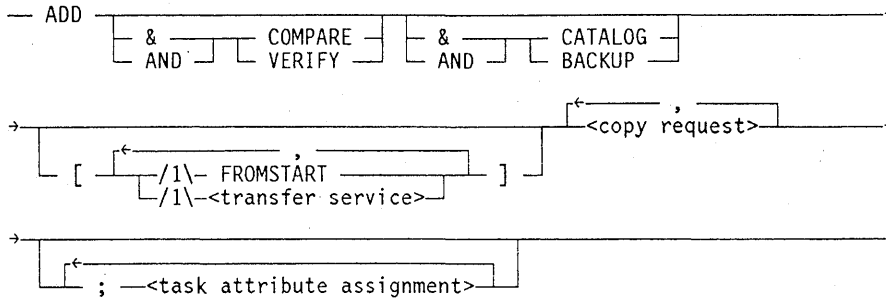
**Examples**

```
ACCESS  
#ACCESSCODE = CODE1.
```

```
ACCESS CODE/PASS  
#SESSION 4094 ET=8:10.1 PT=1:08.9 IO=3.4.1  
#CONTINUE SESSION 4094 13:32:47 10/17/83  
#ACCESSCODE = CODE.
```

# ADD

## Syntax



## Explanation

The ADD command is similar to the COPY command. It copies files between disks and tapes. For a more detailed explanation and the complete syntax, see the COPY command in this section.

The ADD command has the following effects based on whether a disk or tape destination is specified:

- For a disk destination, the ADD command only copies those files that are not already resident on the specified disk destination.
- For a tape destination, the ADD command has the same effect as a COPY command that specifies a tape destination.

The ADD command is particularly useful for adding a directory of files to a disk where some of the files are already resident and are to be preserved.

## Example

The following example copies files under the directory Z/= from tape T to disk R and to DISK. Any files already resident on the destination volumes are not copied. Note that different files might be copied to R and DISK, depending on what is already resident on each destination volume before the ADD is done.

```
ADD Z/= FROM T(KIND=TAPE) TO R(KIND=DISK), TO DISK;
```

# ALTER

## Syntax

```
ALTER <file title> <file attribute assignment>
```

Diagram showing the syntax: ALTER is followed by a bracketed section containing <file title> and <directory name>, followed by a comma and <file attribute assignment>.

### <file attribute assignment>

```
( /1\ APL = <Boolean expression>
  /1\ LOCKEDFILE = <Boolean expression>
  /1\ NOTE = <string>
  /1\ RELEASEID = <string>
  /1\ SAVEFACTOR = <number>
  /1\ SECURITYGUARD = <file title>
  /1\ SECURITYTYPE = <file mnemonic primary>
  /1\ SECURITYUSE = <file mnemonic primary>
  /1\ SENSITIVEDATA = <Boolean expression>
  /1\ USERINFO = <integer> )
```

Diagram showing the syntax for <file attribute assignment>: A list of attributes in parentheses, each with a slash and backslash, followed by an equals sign and a value. The attributes are APL, LOCKEDFILE, NOTE, RELEASEID, SAVEFACTOR, SECURITYGUARD, SECURITYTYPE, SECURITYUSE, SENSITIVEDATA, and USERINFO.

## Explanation

The ALTER command changes the file attributes of a disk file.

Table 3-1 shows the file attributes that can be changed through the ALTER command.

**Table 3-1. File Attributes for the ALTER command**

File Attribute	Meaning
APL	When set to TRUE, this file attribute specifies that only a program whose code file also has its APL attribute set to TRUE can access the file. If the APL attribute is set to FALSE, the file can be accessed by a program regardless of the APL attribute value in the program code file.
LOCKEDFILE	When set to TRUE, this file attribute prevents disk files from being removed or replaced, and the file name from being changed. However, the locked file can be opened and updated, and the file attributes can be changed. When set to FALSE, this attribute permits files to be removed and changed.
NOTE	Stores a message of up to 250 characters to be printed on the banner page preceding the printer file, punch file, or disk file. The default value is a null string.
RELEASEID	Specifies or determines the Mark release level of the file.
SAVEFACTOR	Indicates the expiration date of a file in terms of the number of days past the creation date.

continued

**Table 3-1. File Attributes for the ALTER command** (cont.)

<b>File Attribute</b>	<b>Meaning</b>
SECURITYGUARD	Identifies the guard file to be invoked for the file if the SECURITYTYPE attribute is assigned GUARDED or CONTROLLED. For more information about guard files, refer to the <i>A Series Security Features Operations and Programming Guide</i> .
SECURITYTYPE	Specifies the usercodes, other than the owner of a file, that can access a physical file. The SECURITYTYPE attribute can have a value of PRIVATE (default), PUBLIC, GUARDED, or CONTROLLED. <ul style="list-style-type: none"> <li>• PRIVATE files can be accessed or overwritten only by their owners and privileged users.</li> <li>• PUBLIC files can be accessed by tasks with any usercode, as limited by the setting of the SECURITYUSE attribute.</li> <li>• GUARDED files can be accessed by the owner, however, nonprivileged users and programs are granted access as defined by the guard file. The guard file, which define the access rights to files, must be examined before access to a disk file is granted.</li> <li>• CONTROLLED files can be accessed after the guard file is examined and access to your disk file is granted. If you are not defined in the guard file, you do not have access to the file.</li> </ul>
SECURITYUSE	Specifies how a physical file that is protected by security can be accessed by nonprivileged users using nonprivileged programs. This attribute can have a value of IO (default), IN, or OUT. When a PUBLIC file is accessed by a task with a usercode that differs from the FAMILYOWNER, the SECURITYUSE attribute permits the following actions based on its value: <ul style="list-style-type: none"> <li>• A value of IO permits reading, writing, overwriting, and purging.</li> <li>• A value of IN permits reading, but not writing, overwriting, or purging.</li> <li>• A value of OUT permits writing, overwriting, or purging, but not reading.</li> </ul>
SENSITIVEDATA	When this file attribute is set to TRUE, it causes the disk or pack areas assigned for a file to be overwritten with an arbitrary pattern before the disk space is returned to the system for reallocation.
USERINFO	Saves site- or application-specific information.

Refer to the *A Series File Attributes Programming Reference Manual* for more information about the file attributes that can be specified in the ALTER command.

**Example**

ALTER FILEX (LOCKEDFILE)

## ALTER Command (cont.)

---

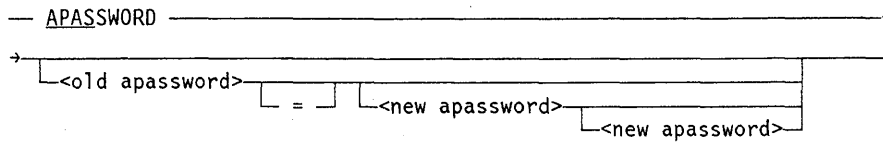
ALTER MYFILE, \*SOURCE/AFILE (LOCKEDFILE=TRUE, SECURITYUSE=IN)

ALTER FILEY, MYFILE/= (SENSITIVEDATA, LOCKEDFILE),  
FILEX (SAVEFACTOR=30, NOTE="Banner Page", LOCKEDFILE)



# APASSWORD

## Syntax



<old apassword>

—<password>

<new apassword>

—<password>

## Explanation

The APASSWORD command allows the accesscode password to be changed for the current accesscode of a session.

To change accesscode passwords, all or part of the required information can be entered with the APASSWORD command. Any information not initially provided is requested by CANDE. For verification purposes, the < new apassword > (new accesscode password) must be entered twice.

To provide complete information, the following would be entered:

```
APASSWORD <old apassword> <new apassword> <new apassword>
```

This form changes the user's accesscode password from the current accesscode password (< old apassword >) to a new accesscode password (< new apassword >).

On InfoGuard systems with the SECOPT CLASS option set to S1 or higher, the old and new passwords must be entered on lines that are separate from the command and from each other.

If APASSWORD or any other subset of the complete command is entered, CANDE prompts the rest of the command.

For further information about accesscode passwords, refer to the "User Identification and Logging On" in Section 1, "General Information."

Any CANDE schedule sessions inherit the accesscode of the CANDE session. The APASSWORD command is not valid within a CANDE schedule session. If a schedule session contains an ACCESS command with an invalid accesscode, the schedule session is aborted.

CANDE includes the accesscode in all log records about a session, particularly in log-on and log-off records.

## APASSWORD Command (cont.)

---

### Examples

```
APASSWORD ESR
#ENTER NEW ACCESSCODE PASSWORD PLEASE.
EPR
#RE-ENTER NEW ACCESSCODE PASSWORD PLEASE.
EPR
#
```

```
APASSWORD OLDPW = NEWPW NEWPW
#
```





## BACKUPPROCESS Command (cont.)

---

### HELP

-----

COPY: Copies a backup file to disk.  
DIRectory: Displays or changes a directory.  
First: Displays the first page of a backup file.  
HELP HELP: Displays the available HELP options.  
LAsT: Displays the last page of a backup file.  
List: Lists a backup file.  
Next: Selects a backup file (same as SELECT).  
OptiOn: Displays or sets program options.  
PRINT: Initiates a print request.  
QUERY: (same as WHAT).  
Quit: Terminates a Backup Processor session.  
REmove: Removes a backup file or directory.  
SAme: Restores the current text page.  
Select: Selects a backup file.  
WHAT: Displays information about a backup file.  
+: Scrolls forward through the file.  
-: Scrolls backwards through the file.

Enter HELP <command> for more information about a command.

%

### WHAT

File # 1: \*BD/0001090/0001100/000OUTFILE ON PACK  
Created on: 09:53 ON 09/01/89  
Job: 1090 "Session"  
Task: 1100 (UZER)"CANDE WRITER"

Print Status:	Not yet queued	Length:	202 lines
PrinterKind:	IP/LP	PrintCopies:	1
Usercode:	(UZER)	TrimBlanks:	True
Chargecode:	9999	Checkpoint:	False
Origin:	TA141/CANDE/3	Trainid:	None
Host:	A15	Banner:	False
Disposition:	E0J	After:	As soon as possible
Save Status:	Remove after printing	Alignment:	False

PCF: Not Specified  
Destination: Not Specified  
FormID: Not Specified  
Transform: Not Specified  
Note: Not Specified  
Alignfile: Not Specified

%

back :rem

#RUNNING 4897

> Remove in progress...

#4897 PK67 \*BD/0004889/0004894/000OUTFILE REMOVED ON PACK

#4897 PK67 \*BD/0004889/0004895/000OUTFILE REMOVED ON PACK

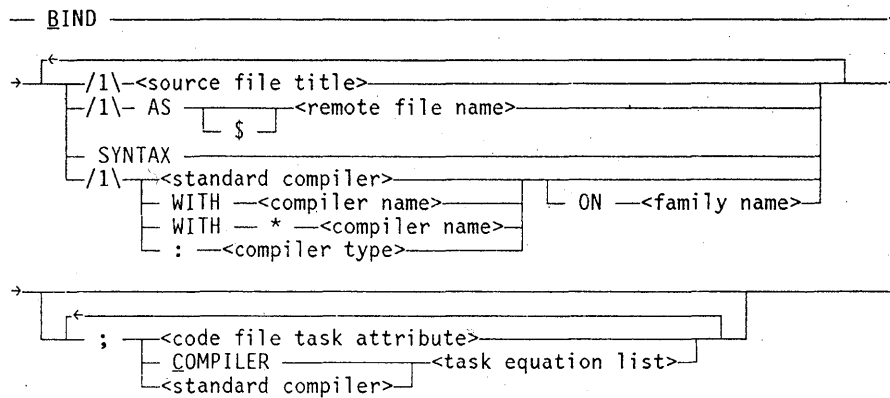
The backup files from SESSION 4889 have been removed.

2 files removed.

#

# BIND

## Syntax



**<source file title>**

—<file title>—

**<remote file name>**

—<file name>—

## Explanation

The BIND command is similar to the COMPILE command except that if a compiler is not specified, the BINDER is invoked. For further information, refer to the COMPILE command in this section.

## Examples

```

BIND
#BINDING 3291
#ET=1:13.9 PT=6.3 IO=60.9
  
```

```

BIND ALGOL/PROG WITH ALGOL
#BINDING 3314
#ET=1:19.6 PT=8.9 IO=50.6
  
```

# BYE

## Syntax

— BYE \_\_\_\_\_%

## Explanation

The BYE command terminates the current CANDE session and disconnects dial-up lines.

If an unsaved work file exists at the time BYE is entered, CANDE displays

```
#REMOVE OR SAVE WORKFILE
```

The previous message indicates the presence of the unsaved work file. The work file must be removed or saved. Then the BYE command must be re-entered before the session will terminate.

System resource usage statistics are provided on termination of the session. The session number, elapsed time of session (ET), processor time consumed (PT), I/O time consumed (IO), usercode, current time, and the date are printed in the following format:

```
#END SESSION <session number> ET=<time> PT=<time> IO=<time>  
#USER = <usercode> <time> <date>
```

If a chargecode is used for the session, then the chargecode is printed following the usercode identification. If an accesscode is used, it is printed following the chargecode.

## Example

```
BYE  
#  
#END SESSION 3003 ET=1:18:19.6 PT=1:08.9 IO=1:02.0  
#USER = UZER 11:27:06 04/20/78
```



If the <new file name> or the associated object file already exists, then the file name or file title specified is not changed. Also, any file names in the directory name or directory title specified that already exist in the <new directory name>, or any object files associated with file names that already exist in the object directory associated with the new directory name are not changed.

If <family name> is specified, the command applies only to the file name or directory name on the volume associated with family name. For example, CHANGE FILEX ON DISK TO FILEZ affects only files found on DISK even though the family specification for the session is FAMILY DISK = USERPACK OTHERWISE OTHERPACK. Refer to "Family Substitution" in Section 1, "General Information" for more information.

If <family name> is not specified, the command applies to the file name or directory name found on DISK. If DISK is also the <target family> in the family specification for the session, then the only files affected by the command are those on the <substitute family>. Files on the <alternate family> are not affected. (Refer to "Family Substitution" in Section 1, "General Information" for more information.)

In a <change from group> construct, a FROM <family name> clause applies to all the file names or directory names in that <change from group>. An ON <family name> clause in the <file title> or <directory title> constructs applies only to the immediately preceding file name. An ON clause cannot precede a FROM clause in the same command.

When specifying a file name, file title, or directory name, a user logged on under a nonprivileged usercode cannot enter a usercode specification other than the usercode for the current session.

If SOURCE or OBJECT is used as a file name, then it must be enclosed in quotation marks to prevent ambiguity with the keywords SOURCE and OBJECT.

If the CHANGE command is used to change the name of a file whose LOCKEDFILE file attribute is set to TRUE, the file name is not changed. CANDE displays the following message to indicate that the file name was not changed:

```
<file name> NOT CHANGED (LOCKEDFILE).
```

For more information about the LOCKEDFILE file attribute, refer to the *File Attributes Reference Manual*. See the ALTER command earlier in this section for information about changing the LOCKEDFILE attribute.

The following paragraphs contain an explanation for each syntactic variation of the CHANGE command.

```
CHANGE <new file name>
CHANGE TO <new file name>
```

CHANGE <new file name> and CHANGE TO <new file name> change the name of the current work file to a new file name. Additionally, if a workobject file exists, its name is changed to become the object file associated with the new file name. If no work file exists, the command is rejected with an error message. Neither a usercode nor an asterisk (\*) is allowed in the new file name.

## CHANGE Command (cont.)

---

**CHANGE <file title> TO <new file name>**  
**CHANGE SOURCE <file title> TO <new file name>**

CHANGE <file title> TO <new file name> changes the name of the file specified by the file title to a new file name. Additionally, if the associated object file exists, its name is changed to the object file name associated with the new file name. If SOURCE is specified, the name of the file specified by the file title is changed to a new file name; any object file associated with the file title is not affected. The dollar sign (\$) in front of a new file name is not allowed.

**CHANGE <directory title> TO <new directory name>**  
**CHANGE SOURCE <directory title> TO <new directory name>**

CHANGE <directory title> TO <new directory name> changes the names of the files in the directory specified by the directory title to the appropriate file names in the new directory name. The command also changes the names of the files in the associated object directory to the object file names associated with new directory name. If SOURCE is specified, the names of the files in the directory specified by the directory title are changed but the associated object directory is not affected. The dollar sign (\$) in front of a new directory name is not allowed.

**CHANGE OBJECT <file title> TO <new file name>**  
**CHANGE OBJECT <file title> TO \$ <new file name>**  
**CHANGE OBJECT \$ <file title> TO <new file name>**  
**CHANGE OBJECT \$ <file title> TO \$ <new file name>**

CHANGE OBJECT <file title> TO <new file name> changes the name of the object file associated with a file title to the name of the object file associated with the new file name. The dollar sign (\$) can precede either the file title, or the new file name, or both. For the file title, the dollar sign indicates that the file is not stored with the associated object file name but is stored as a file title. For the new file name, the dollar sign indicates that the changed file receives a new file name as its name instead of the object file name associated with the new file name.

**CHANGE OBJECT <directory title> TO <new directory name>**  
**CHANGE OBJECT <directory title> TO \$ <new directory name>**  
**CHANGE OBJECT \$ <directory title> TO <new directory name>**  
**CHANGE OBJECT \$ <directory title> TO \$ <new directory name>**

CHANGE OBJECT <directory title> TO <new directory name> changes the names of the files in the object directory associated with the directory title to the appropriate file names in the object directory associated with the new directory name. The dollar sign (\$) can precede either the directory title, or the new directory name, or both. For the directory title, the dollar sign indicates that the directory is not stored as the associated object directory name but is stored as a directory title. For the new directory name, the dollar sign indicates that the files in the changed directory receive a new directory name as their directory name instead of the object directory name associated with the new directory name.



**Examples**

The following example renames the work file from A to B:

```
WHAT
#WORKFILE A: ALGOL
TITLE B
#
WHAT
#WORKFILE B: ALGOL
```

The following example renames the source file A to B and the object file OBJECT/A to OBJECT/B:

```
TITLE A B
#(UZER)A ON USERPACK (& OBJECT) CHANGED TO (UZER)B
```

The following example renames the source file A, the source file B, and the object file OBJECT/B to A1, B1, and OBJECT/B1, respectively. All files reside on the family OTHERPACK.

```
CHANGE SOURCE A TO A1, B TO B1 FROM OTHERPACK
#(UZER)A ON OTHERPACK CHANGED TO (UZER)A1
#(UZER)B ON OTHERPACK (& OBJECT) CHANGED TO (UZER)B1
```

The following example renames the file A to B on the family USERPACK:

```
CHANGE A ON USERPACK TO B
#(UZER)A ON USERPACK CHANGED TO (UZER)B
```

The following example renames the object file associated with the file TEST (named OBJECT/TEST) to TEST:

```
CHAN OBJECT TEST TO $TEST
#(UZER)OBJECT/TEST ON USERPACK CHANGED TO (UZER)TEST
```

The following example renames the object file TESTONLY (which is not stored under the object directory) to OBJECT/TESTONLY:

```
CHAN OBJECT $TESTONLY TO TESTONLY
#(UZER)TESTONLY ON USERPACK CHANGED TO (UZER)OBJECT/TESTONLY
```

## CHANGE Command (cont.)

---

The following example renames the files of the directory PROGRAMLIB and its associated object directory:

```
FILE PROGRAMLIB
(UZER) ON USERPACK
. PROGRAMLIB
. . A : ALGOL
. . B : ALGOL
#
```

```
FILE LIB
#NO FILE(S) ON USERPACK
#
```

```
FILE OBJECT
(UZER) ON USERPACK
. OBJECT
. . LIB
. . . B : FORTRANCODE
. . PROGRAMLIB
. . . A : ALGOLCODE
#
```

```
TITLE PROGRAMLIB/= LIB/=
#2 FILES IN (UZER)PROGRAMLIB/= CHANGED TO (UZER)LIB/= ON USERPACK
#1 FILE IN (UZER)OBJECT/PROGRAMLIB/= CHANGED TO (UZER)OBJECT/LIB/=
ON USERPACK
```

```
FILE PROGRAMLIB
#NO FILE(S) ON USERPACK
#
```

```
FILE LIB
(UZER) ON USERPACK
. LIB
. . A : ALGOL
. . B : ALGOL
#
```

```
FILE OBJECT
(UZER) ON USERPACK
. OBJECT
. . LIB
. . . A : ALGOLCODE
. . . B : FORTRANCODE
#
```

# CHARGE

## Syntax

```

— CHARGE <chargecode>

```

```

<chargecode>

```

```

— <name>

```

## Explanation

The CHARGE command specifies a new chargecode at some point in a session.

When CHARGE is entered, the chargecode reverts to a null chargecode specification unless USEDEFAULTCHARGE and CHARGEREQ, or just CHARGEREQ, are SET in the USERDATAFILE entry for that usercode. If both USEDEFAULTCHARGE and CHARGEREQ are SET, transmitting CHARGE causes the chargecode to revert to the default chargecode. If CHARGEREQ (but not USEDEFAULTCHARGE) is SET in the USERDATAFILE for the current usercode, and CHARGE is entered, an error message is given.

When CHARGE <chargecode> is entered, the chargecode changes to the specified <chargecode>. If CHARGEREQ is SET in the USERDATAFILE, <chargecode> must be validated in the USERDATAFILE. If CHARGEREQ is not SET, any chargecode can be entered; validation in the USERDATAFILE is not necessary. Specifying a new chargecode provides a breakdown of system usage statistics incurred under the previous chargecode. When a new chargecode is entered, the system usage statistics for the session are set to zero as if the previous session had terminated and a new one had begun.

For more information on chargecodes, refer to “User Identification and Logging On” in Section 1, “General Information.” Refer to the *Security Administration Guide* for further information on the USERDATAFILE.

## Examples

```

charge
#SESSION 3601 ET=9.0 PT=0.0 IO=0.0
#CONTINUE SESSION 3601 14:08:03 4/20/78

```

```

CHAR ME
#SESSION 3601 ET=7:47.2 PT=55.8 IO=57.1
#CONTINUE SESSION 3601 14:15:50 04/20/78

```

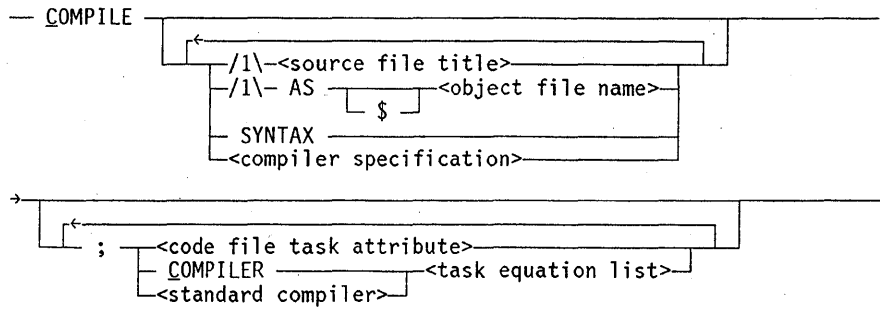
```

CHAR 12345678901234567
#SESSION 3601 ET=5:08.9 PT=0.5 IO=0.2
#CONTINUE SESSION 3601 14:20:59 04/20/78

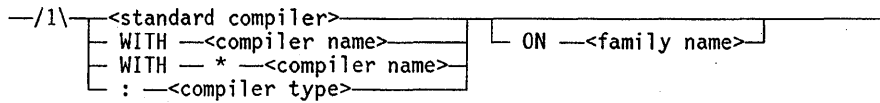
```

# COMPILE

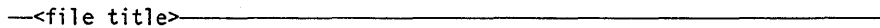
## Syntax



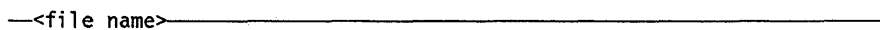
### <compiler specification>



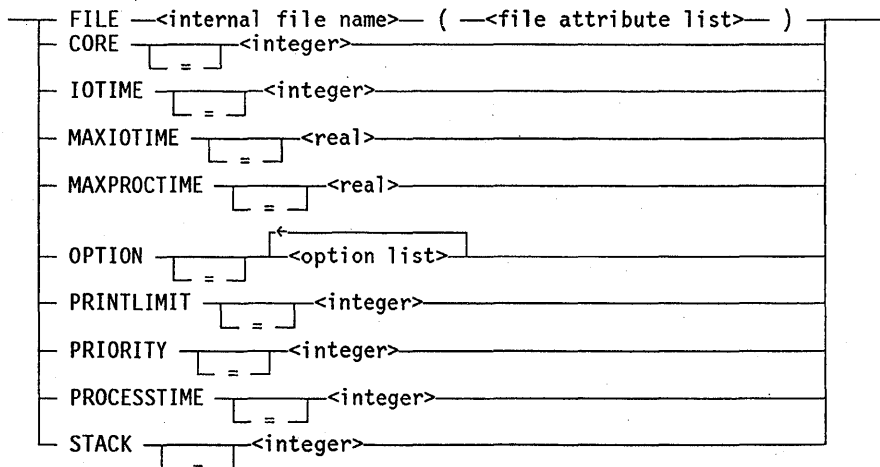
### <source file title>



### <object file name>



### <code file task attribute>



## Explanation

The COMPILE command invokes a compiler to compile a program, and if there are no syntax errors in the program, the result is an object program.

**Specifying a file to be compiled**

You can specify the title of the source file that is to be compiled; however, if a source-file title is not specified, the work file is assumed. If the file to be compiled is the work file, an UPDATE operation is performed before the compiler is invoked. The following examples show a COMPILE command that specifies a source-file title, and a COMPILE command that assumes the work file, respectively:

```
COMPILE MY/ALGOL/PROGRAM
```

```
COMPILE
```

If a source-file title is specified, the resulting object program is titled OBJECT/<source file title>. If the work file is specified or assumed and the work file is saved before it is compiled, then the object program is titled OBJECT/<work file title>. If an object file name other than the source-file title is desired, then the following COMPILE syntax should be used:

```
COMPILE <source file title> AS OUR/ALGOL/PROGRAM
```

If an object file name other than the work file title is desired, then either of the following COMPILE statements can be used, resulting in an object program titled OBJECT/OUR/ALGOL/PROGRAM:

```
COMPILE <work file title> AS OUR/ALGOL/PROGRAM
```

```
COMPILE AS OUR/ALGOL/PROGRAM
```

If a dollar sign (\$) precedes the object file name, the OBJECT directory is omitted. An asterisk (\*) or usercode is not permitted with an object file name.

**Specifying a compiler**

A particular compiler can be specified to compile a source file or work file; however, if a compiler is not specified, the compiler appropriate for the source-file type is invoked. For example, the ALGOL compiler is invoked for type ALGOL files. The following are examples of the COMPILE command with a compiler name specified:

```
COMPILE MY/ALGOL/PROGRAM ALGOL
```

```
COMPILE HIS/COBOL74/PROGRAM C74 ON PACK
```

If a compiler is not specified and the type of the source file or work file is not a compiler type (such as SEQ or DATA), CANDE displays one of the following messages:

```
#SOURCE NOT COMPILER TYPE
```

```
#WORKFILE NOT COMPILER TYPE
```

Nonstandard compilers (such as BDMSCOBOL and BDMSALGOL) can be invoked by using the WITH construct.

## COMPILE Command (cont.)

---

Because the prefix *SYSTEM/* is assumed to precede the first node of the compiler title after any usercode or asterisk (\*), the prefix *SYSTEM/* should not be explicitly specified. The compiler title has the same form as a file title, except that the maximum number of nodes is 11. For example, the following command uses *SYSTEM/ALGOL* compiler:

```
COMPILE PROGRAM/X WITH ALGOL
```

Some of the older (prior to Mark 3.1) compilers have been changed in their use of internal file names to make them consistent with the newer compilers. As a result of this change, the new internal file name *ERRORS* should be used instead of *ERRORFILE* when an *ALGOL*, *COBOL*, *COBOL74*, *DASDL*, or *NEWP* compilation is started from *CANDE* with a file equation for the error file. In addition, if you do not supply a value for the *TITLE* file attribute, but you do set the *KIND* file attribute to *DISK* and the *PROTECTION* file attribute to *SAVE*, then the error file is created under the name *ERRORS* rather than *ERRORFILE*.

If you use the *COMPILE* command with the compiler specification syntax of *COMPILE <file> WITH \* <compiler name>*, then *CANDE* uses the compiler *\*SYSTEM/<compiler name>*.

If you use the *COMPILE* command with the compiler specification syntax of *COMPILE <file> : <compiler type>*, then *CANDE* requires the short form notation of the compiler name. For example, *C74* must be the compiler type used instead of *COBOL74*. The valid compiler types are listed in this manual under *<standard compiler>* in Section 2, "Basic Constructs".

### Compiling for syntax only

The *SYNTAX* option informs the compiler to compile for syntax only. No object file results when this option is specified.

When syntax errors are found, they are displayed. Syntax error messages can be sent to a backup file by including the *\$SET LIST* record in the program to be compiled.

### Compiling with a specified compiler task equation list

Task attributes can be specified for the compile task. The following is an example of a task attribute that modifies the compile task and results in the compiler running with a priority of 30:

```
COMPILE; COMPILER PRIORITY=30
```

The following example results in the file *ERRORS* declared by the compiler to be equated to *KIND=PRINTER*.

```
COMPILE; COMPILER FILE ERRORS (KIND=PRINTER)
```

### Compiling with code file task attributes specified

Code file task attributes specified in the *COMPILE* statement get stored in the code file after a successful compile. These attributes take effect when the code file is executed.

The following example shows the program UTIL/ALGOL/EXAMPLE which is to be compiled by the ALGOL compiler. The code-file task equation in the example equates the task attribute MAXPROCTIME to 50 seconds.

```
GET UTIL/ALGOL/EXAMPLE; C WITH ALGOL; MAXPROCTIME = 50
```

### Examples

```
c  
#COMPILING 3673  
#ET=7.2 PT=0.4 IO=0.4
```

```
COMPILE WITH BDMSCOBOL  
#COMPILING 3698  
#3721 BOJ *DATABASE/INTERFACE ON USERPACK  
#3721/3721 EOJ JOB DATABASE/INTERFACE ON USERPACK  
#ET=2:01.1 PT=1:03.8 IO=12.9
```

```
c a/b as $test/a/b:dcalgol  
#COMPILING 3725  
#ET=29.8 PT=5.6 IO=2.3
```

```
COMPILE A ON B AS Z WITH MYCOMP ON MYPACK  
#COMPILING 3727  
#ET=1:28.6 PT=46.3 IO=36.4
```





# CONNECT

## Syntax

```

CONNECT TO <hostname> : <MCS name>
<MCS name>
<file title>

```

## Explanation

The CONNECT command invokes the Station Transfer capability of BNA. Station Transfer allows a user to transfer control of a data comm station from the MCS of a local host to an MCS on a remote host. Once the logical connection is established between the station and the remote MCS, the station behaves as if it were actually connected to the remote host.

The CONNECT command initiates a station transfer dialog between the local host and the host specified by <hostname>.

If an <MCS name> is given and the STATIONTRANSFER MCS restriction is not set on the remote host, the station is transferred to the MCS on the remote host. If no <MCS name> is specified, the station is transferred to the default MCS of the remote host.

If the MCS specified by <MCS name> is not running, an attempt will be made to start it when the station is transferred to it.

The <MCS name> may not be SYSTEM/STATION/TRANSFER.

If an <MCS name> other than the specified default MCS is specified and the STATIONTRANSFER MCS restriction is set on the remote host, the CONNECT operation fails, and control of the station is returned to CANDE.

Once the station has been transferred to the remote MCS, that MCS may require the user to log on with a valid usercode/password for that host.

The BNA network must be running at the local host before a Station Transfer dialog can be initiated. If the local host is not in network operating mode, the CONNECT will fail and control of the station will return to CANDE.

The host that is specified by <hostname> must also be in network operating mode for a successful connection.

For the Station Transfer feature, at least one pseudostation must exist on the host receiving the transferred station; the number of pseudostations limits the number of remote stations that can be transferred in. A pseudostation is a virtual station that can be attached to, and controlled by, an MCS in a manner similar to a *real* station. However, unlike a real station, a pseudostation is not declared in NDLII, has no line assigned, and need not have a corresponding physical terminal on the local host.

## CONNECT Command (cont.)

---

### Examples

CONNECT TO HOSTA

CONNECT TO HOSTB: SYSTEM/CANDE

# CONTINUE

## Syntax

```

— CONTINUE [ ? | * | = <special character> | <control char mnemonic> ]

```

## Explanation

CANDE allows lines of input to be continued through the CONTINUE command. The user can reset, interrogate, or define a continuation character with the CONTINUE command.

CANDE interprets a line of normal input (that is, a line not beginning with the control character for the user's station) as needing to be continued when the defined continuation character is the last character of input. It removes the continuation character and inserts one blank in its place. CANDE sends `#%` as a continuation prompt. The next line of input is then appended to this line. CANDE does not continue a line of normal input entered in sequence mode or page mode. (CANDE treats a defined continuation character entered in page or sequence mode as text.)

If CONTINUE is entered, the continuation character becomes undefined. If *CONTINUE ?* is entered, the user's current continuation character is displayed.

Entering *CONTINUE \** sets the continuation character to the default value defined for it in the USERDATAFILE. If the default continuation character is not defined there, the tasking-only continuation character (`%`) is recognized as the default.

Entering either *CONTINUE = <special character>* or *CONTINUE = <control char mnemonic>* defines a new continuation character. Any special character except the semicolon (`;`) can be used for the continuation character.

Each user can have a default continuation character defined in the USERDATAFILE. The CANDECONTCHAR option is initialized at the start of each CANDE session as the default continuation character. The user can override the default character during a particular session by using the CONTINUE command. Refer to the *Security Administration Guide* for further information about the USERDATAFILE.

If no default continuation character is defined in the USERDATAFILE, the ability to continue task-type commands with a percent sign (`%`) exists until the first use of any form of the CONTINUE command except for *CONTINUE ?*.

If this system default continuation character is in effect, a query of the continuation character results in the "TASKING-ONLY CONTINUATION CHARACTER (`%`) DEFINED" message.

## CONTINUE Command (cont.)

---

### Examples

```
CONT
#NO CONTINUATION CHARACTER DEFINED
```

```
CONT *
#TASKING-ONLY CONTINUATION CHARACTER (%) DEFINED
```

```
CONT = *
#CONTINUATION CHARACTER = *
```

```
TERM WIDTH 28
#LINE = 28, PAGE = 24, BUF =1920, SCREEN, WAIT, "\"
```

```
MAKE*
#%
A*
#%
ALGOL
#WORKFILE A: ALGOL
100 ENTER A LONG LINE ON A*
#%
NARROW SCREEN
```

```
LIST
100 ENTER A LONG LINE ON\
\A NARROW SCREEN
#
```

```
CONT
#
```

```
CONT ?
#NO CONTINUATION CHARACTER DEFINED
```



## CONVENTION Command (cont.)

---

### Examples

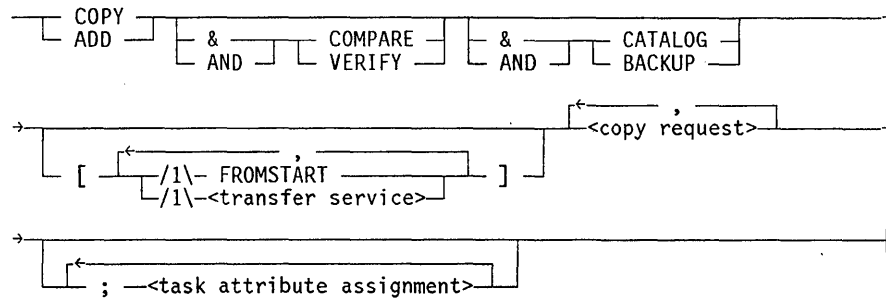
```
CONVENTION  
#CONVENTION: AMERICAN
```

```
CONVENTION CANADIAN  
#CONVENTION CHANGED FROM AMERICAN TO CANADIAN
```

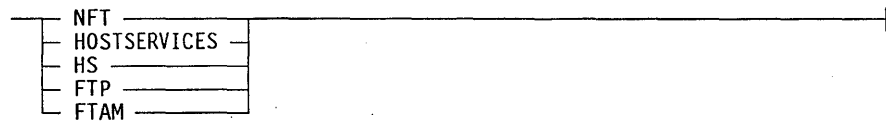
```
CONVENTION *  
#CONVENTION CHANGED FROM CANADIAN TO AMERICAN
```

# COPY or ADD

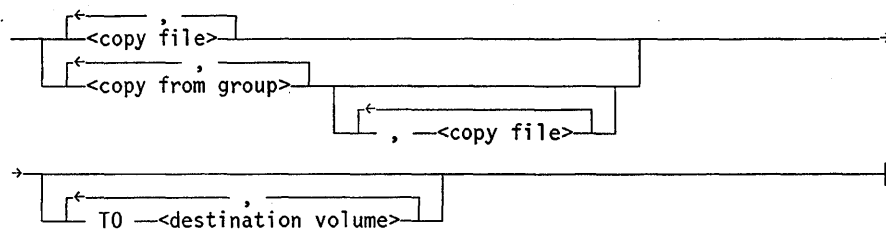
## Syntax



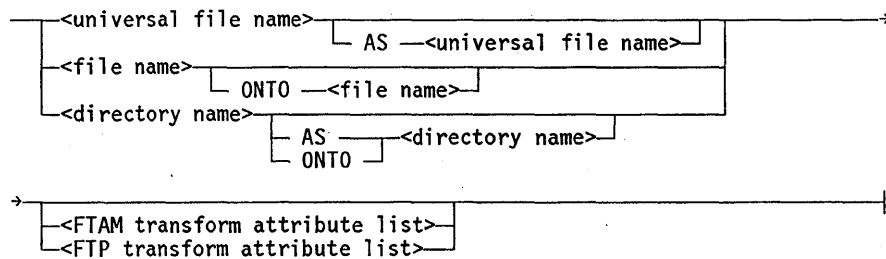
### <transfer service>



### <copy request>



### <copy file>

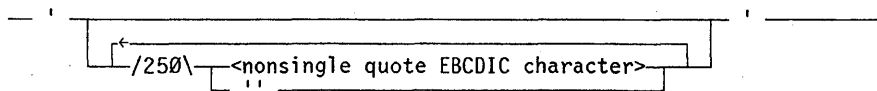


### <universal file name>

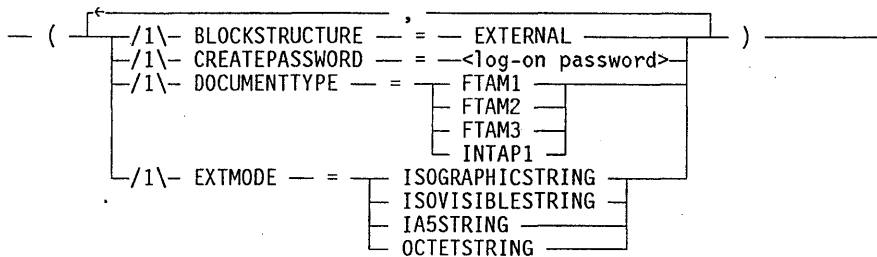


## COPY Command (cont.)

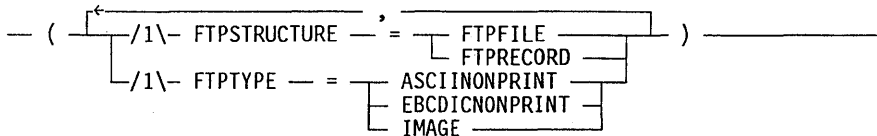
### <interchange name>



### <FTAM transform attribute list>



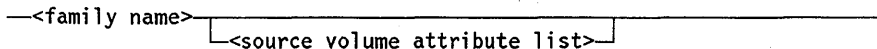
### <FTP transform attribute list>



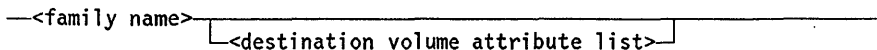
### <copy from group>



### <source volume>

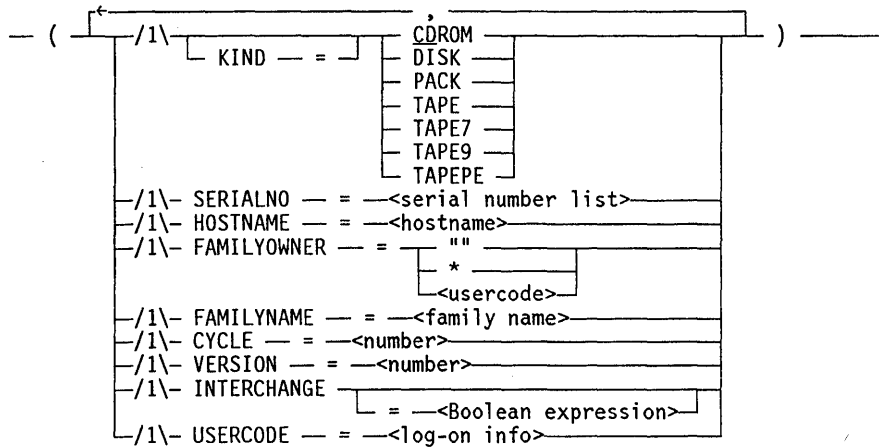


### <destination volume>

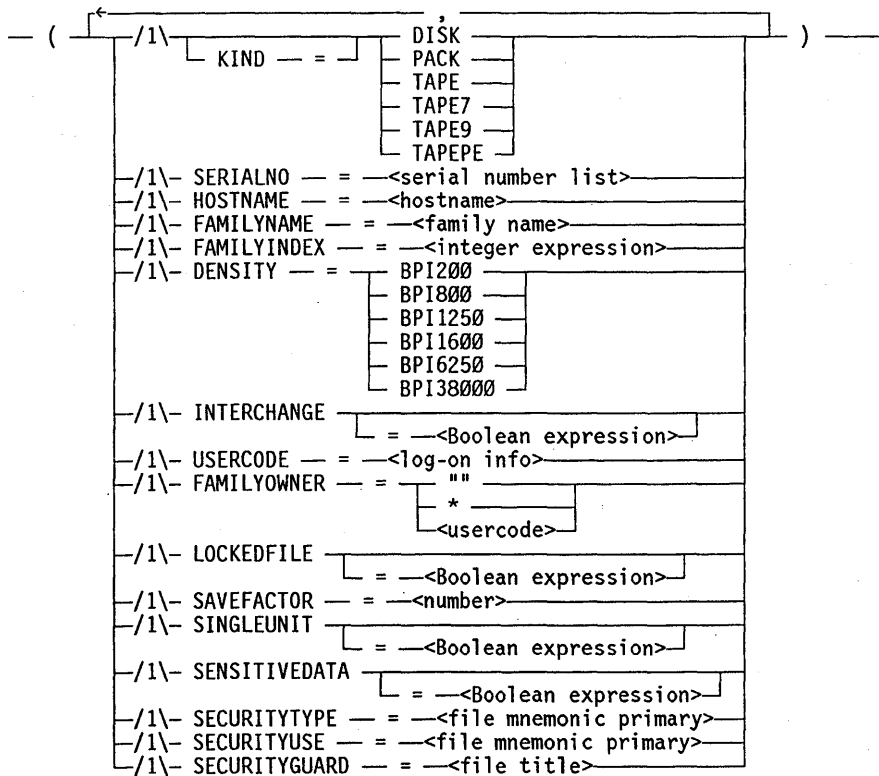




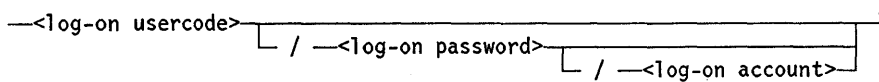
<source volume attribute list>



<destination volume attribute list>



<log-on info>



## COPY Command (cont.)

---

<log-on usercode>  
<log-on password>  
<log-on account>

```
|<name>|-----|
|<interchange name>|
|  "  |
|  "  |
|  "  |
```

### Explanation

The COPY command copies files between disks and tapes. The ADD command is similar to the COPY command. The ADD command has the following effects based on whether a disk or tape destination is specified:

- For a disk destination, the ADD command only copies those files that are not already resident on the specified disk destination.
- For a tape destination, the ADD command has the same effect as a COPY command that specifies a tape destination.

The ADD command is particularly useful for adding a directory of files to a disk where some of the files already reside and are to be preserved.

For all the following cases, unless otherwise mentioned, explanations about the COPY command also apply to the ADD command.

A COPY or ADD command running under a nonprivileged usercode cannot copy a directory of files that resides under another usercode.

The COPY and ADD commands are processed by the Work Flow Language (WFL), not CANDE. The command (and any text appearing on the same line) is passed unedited to WFL. Additional WFL statements may follow on the same line, but additional CANDE commands may not. WFL uses the normal task message facilities to report the result. As a result, messages related to the success or failure of the request are not seen unless they are requested (through the ?MSG command) or the MESSAGE option is SET. (Refer to the SO command for more information.)

**Note:** *Because CANDE can run with different release versions of the MCP and other software, it is possible that a feature of software might not work as described in this manual. For example, if the Mark 4.0 version of CANDE is run with the Mark 3.9 version of the MCP and WFL software, then an attempt to use a Mark 4.0 COPY command option that is passed to WFL, as documented in the Mark 4.0 version of this manual, might cause an error or unexpected result.*

### COMPARE Option

The COMPARE option compares the copied file and the original file bit for bit, immediately after the file is copied. The COMPARE option is not available for quarter inch cartridge (QIC) tape. Refer to the VERIFY option to check the integrity of the copied file on a QIC tape. When certain I/O errors occur or when COMPARE errors are

detected, then a "RECOPY REQUIRED" response request (RSVP message) is issued. The RECOPY condition does not occur when QIC tape is used.

#### VERIFY Option

The VERIFY option causes output files to be read and tested for accuracy with a checksum algorithm. The verification of files for a disk destination takes place immediately for each file after the file is written to disk. When an I/O error or a checksum mismatch occurs during a disk file verification, the following message is issued:

```
RECOPY REQUIRED
```

The verification of files for a tape destination takes place after the entire output tape volume is written. When an I/O error or checksum mismatch occurs for a tape destination, the following RSVP message is issued:

```
MTnn VERIFY ERROR; REPLY 'FR' TO PURGE TAPE, 'OF' TO CONTINUE WITH  
NEXT FILE <file name>
```

You can then decide to either terminate the copy and purge the tape, or ignore the error and continue with the copy.

Source files that are copied from tape can also be verified. If the source tape was created with the VERIFY option, the checksums of the file from the source tape can be compared with the checksums of the copied file. If the source tape was created without the VERIFY option, that is, without checksums, then the source checksum testing is bypassed and the following message is issued:

```
WARNING, TAPE WAS NOT WRITTEN WITH CHECKSUMS
```

#### BACKUP and CATALOG Options

The BACKUP and CATALOG options apply only to a cataloged system. The BACKUP option causes backup copies of the cataloged files to be created. The CATALOG option causes the copied files to be entered in the cataloged system. The destination volume must be added to the volume library by the WFL *VOLUME ADD* statement before the files can be cataloged.

#### FROMSTART and <transfer service> Options

The FROMSTART option takes effect when you are reissuing a previously aborted file transfer. With this option, you can specify whether to have all files named in the COPY command transferred again.

The <transfer service> construct is used to specify one of four file transfer services to use between hosts:

- A Series Native File Transfer (NFT)
- Host Services File Transfer (HOSTSERVICES or HS)
- File Transfer Protocol (FTP)

- File Transfer, Access, and Management (FTAM)

The transfer service used is dependent on the hosts and the type of file that is to be transferred. More information about file transfer services and the FROMSTART option is given later in this description of the COPY command.

<copy request> and <copy file>

You can specify the name of a file or directory to be copied. Use the <copy file> option of the <copy request> construct to specify the file or directory name.

If you use the AS option of the <copy file> construct and specify a universal file name, the new copy of the file is given the specified file name. If you copy files locally (through library maintenance), the interchange file name option of the universal file name is not valid; the file name must refer to a file resident on an A Series system.

If you use the AS option and specify a directory name, the new copy of each file is renamed with the new directory name and the portion of the file name that follows the old directory name is retained.

If you use the AS option and specify a file or directory name, but do not include a usercode or asterisk (\*) with the new file or directory name, then the new file name is prefixed with the usercode under which the copy process is running.

The ONTO option is intended for use only with Installation Allocated Disk (IAD). For details, refer to the discussion of IADMPPER in the *A Series System Software Utilities Operations Reference Manual*. This option is valid only for data files whose integer value of the FILEKIND mnemonic file attribute is greater than or equal to 64 (ALGOLSYMBOL). The ONTO option is not allowed for code files, compilers, or system files. If the CLASS option of the SECOPT (SECURITY OPTIONS) system command is set to the value S2, then the ONTO option is not allowed for any file. The SECOPT command is available only through InfoGuard software: the Unisys security-enhancement software.

You can also specify FTAM transform attributes in the <copy file> construct to assign certain attributes to files that are transferred to a remote host. The FTAM transform attributes are assigned to the transferred file only. The A Series source file need not reflect nor is it affected by the FTAM transform attributes assigned to the copied file. If an FTAM transform attribute is specified in a COPY statement from a remote host, the following warning is displayed and the FTAM transform attribute is ignored:

WARNING: FTAM ONLY SUPPORTS A <FTAM TRANSFORM ATTRIBUTE> FOR COPIES "TO" A REMOTE HOST.

The FTAM file attributes are BLOCKSTRUCTURE, CREATEPASSWORD, DOCUMENTTYPE, and EXTMODE. These attributes are described later under "Specifying File Attributes" for the COPY command.

You can also assign FTP transform attributes to files that are transferred to or from a remote host through the <copy file> construct. The FTP transform attributes FTPSTRUCTURE and FTPTYPE are described later in this section under "Specifying File Attributes."

**<copy from group>**

In the **<copy from group>** construct, the FROM clause applies to the list of file names and directory names in that **<copy from group>**. It is best to group files to be copied from the same tape volume into one **<copy from group>**, because if the same tape volume is specified more than once, the tape is rewound for each **<copy from group>**.

If the source or destination volume is not specified, DISK is assumed. A source or destination volume is required unless all the **<copy file>** constructs in the command contain an AS name. If more than one destination volume is specified, all files are copied, at the same time and in the same order, to all destination volumes.

If the source or destination is the family named DISK or the family named PACK, the default of KIND is DISK; otherwise, the default of KIND is TAPE.

**<source volume> and <destination volume>**

The **<source volume>** construct includes the family name and the source volume attribute list. The source volume attributes can be specified to identify those files being copied from the source host.

The **<destination volume>** construct includes the family name and the destination volume attribute list. The destination volume attributes assign file characteristics such as location, ownership, security, and so on to the files that are transferred to a remote host.

Refer to "Specifying File Attributes," later in this section, for more information about the source volume and destination volume file attributes.

For the definition of the **<Boolean expression>**, **<file mnemonic primary>**, **<integer expression>**, **<serial number list>**, or **<task attribute assignment>** constructs, refer to the *WFL Reference Manual*. Other constructs are defined there or in Section 2, "Basic Constructs," of this manual.

**<FTAM transform attribute list> and <FTP transform attribute list>**

The FTAM transform attributes are used to assign file characteristics to files that are transferred to a remote host through the FTAM file transfer service. The FTP transform attributes are similarly used for files that are transferred to a remote host through the FTP file transfer service.

Refer to "Specifying File Attributes," later in this section, for more information about the FTAM and FTP transform file attributes.

**<log-on info>**

Use the **<log-on info>** construct to provide information for the source volume attribute or destination volume attribute USERCODE. The log-on information is made up of three optional pieces of information: the log-on usercode, the log-on password, and the log-on account information. Each piece of information is separated by a slash (/).

Each part of the log-on information can contain the following:

- Two quotation marks ( " " ) that indicate no item is being sent.
- Two apostrophes ( ' ' ) that indicate that a string with a length of 0 (zero) is being sent.
- An identifier that is from 1 to 249 characters long and contains any combination of EBCDIC characters other than an apostrophe or a character that has a hexadecimal code less than 4"40". To send an apostrophe to the remote host, the identifier must be enclosed in apostrophes and a single apostrophe must be represented by two apostrophes in a row. To send lowercased characters, the identifier must be enclosed in apostrophes. The enclosing apostrophes are removed before the characters are sent to the remote host.

## Specifying File Attributes

When you use the COPY command, you can indicate certain file attributes for files being copied from a source volume or to a destination volume. The file attributes are specified in the source volume attribute list and the destination volume attribute list.

Table 3-2 shows the file attributes that you can specify in the COPY command along with a brief description of each attribute. The CYCLE and VERSION attributes have no effect as destination volume attributes. For more specific information on an attribute, refer to the *A Series File Attributes Programming Reference Manual*.

**Table 3-2. File Attributes for the COPY Command**

File Attribute	Meaning
CYCLE	Designates the specific generation of a permanent file. This file attribute is used in conjunction with the VERSION attribute.
FAMILYNAME	Indicates the name or label of the disk family on which the physical file is located.
FAMILYOWNER	Indicates the owner of a tape volume family. If you specify a usercode with the FAMILYOWNER attribute, the usercode becomes the owner. If you do not use the FAMILYOWNER attribute or you specify a null string (""), the usercode portion of the universal file name or directory name is used. If you specify an asterisk (*) with the FAMILYOWNER attribute, the asterisk (*) usercode becomes the owner of the tape volume.
HOSTNAME	Indicates a remote host where the source or destination volume is located. The HOSTNAME attribute can be specified for either the source or destination volume, to copy files from or to a foreign host; or it can be specified for both source and destination volumes, to allow files to be copied from one foreign host to another.

continued

Table 3-2. File Attributes for the COPY Command (cont.)

File Attribute	Meaning
INTERCHANGE	Indicates that the file is stored on a removable disk pack that is compatible across certain Unisys systems. This file attribute is valid only when the file KIND is DISK or PACK. Refer to interchange packs in the <i>I/O Subsystem Programming Guide</i> for further details.
KIND	Describes the peripheral unit associated with the logical file.
SERIALNO	Identifies the specific disk or tape volumes to be used when copying files. For more information, refer to the <i>WFL Reference Manual</i> .
USERCODE	Passes log-on information, which consists of a usercode, password, and charge account, to the remote host specified with the HOSTNAME attribute. The log-on information has a maximum length of 255 characters, which includes all quotation marks, single quotation marks, and slashes. The USERCODE attribute is ignored if the HOSTNAME attribute is not specified, the Host Services File Transfer is used, or A Series NFT is used.  YOURUSERCODE is an acceptable synonym for USERCODE.
VERSION	Designates the successive iteration of the same generation of a permanent file. This file attribute is used in conjunction with the CYCLE attribute.

Table 3-3 shows the additional file attributes that can be specified only for the destination volume attribute list. These file attributes cannot be specified for the source volume attribute list.

Table 3-3. File Attributes for the Destination Volume Attribute List

File Attribute	Meaning
DENSITY	Indicates the recording density of a magnetic tape file. For further information about magnetic tape files, refer to the <i>I/O Subsystem Programming Guide</i> .
FAMILYINDEX	Designates a specific physical unit within a disk family. If you do not specify the FAMILYINDEX attribute, the FAMILYINDEX volume of the source volume is used. If you specify a value of 0 or an integer expression that evaluates to 0 for the FAMILYINDEX attribute, then the disk areas are allocated in the normal rotational order of the system.
LOCKEDFILE	When used in the COPY command, the LOCKEDFILE file attribute can be set for files copied to tape only. It has no effect on disk files. If the LOCKEDFILE file attribute is set to TRUE for a file, the entire contents of the specified library tape can be purged only with confirmation from the operator.

continued

Table 3-3. File Attributes for the Destination Volume Attribute List (cont.)

File Attribute	Meaning
SAVEFACTOR	Indicates the expiration date of a file in terms of the number of days past the creation date. The default value for the SAVEFACTOR attribute when a tape is created is 30 days.
SECURITYGUARD	Identifies the guard file to be invoked for the file if the SECURITYTYPE attribute is assigned GUARDED or CONTROLLED. For more information about guard files, refer to the <i>A Series Security Features Operations and Programming Guide</i> .
SECURITYTYPE	<p>Specifies the usercodes, other than the owner of a file, that can access a physical file. The SECURITYTYPE attribute can have a value of PRIVATE (default), PUBLIC, GUARDED, or CONTROLLED.</p> <ul style="list-style-type: none"> <li>• PRIVATE files can be accessed or overwritten only by their owners and privileged users.</li> <li>• PUBLIC files can be accessed by tasks with any usercode, as limited by the setting of the SECURITYUSE attribute.</li> <li>• GUARDED files can be accessed by the owner; however, nonprivileged users and programs are granted access as defined by the guard file. The guard file, which defines the access rights to files, must be examined before access to a disk file is granted.</li> <li>• CONTROLLED files can be accessed after the guard file is examined and access to your disk file is granted. If your usercode is not defined in the guard file, you do not have access to the file.</li> </ul>
SECURITYUSE	<p>Specifies how a physical file that is protected by security can be accessed by nonprivileged users using nonprivileged programs. This attribute can have a value of IO (default), IN, or OUT. When a PUBLIC file is accessed by a task with a usercode that differs from the FAMILYOWNER, the SECURITYUSE attribute allows the following actions based on its value:</p> <ul style="list-style-type: none"> <li>• A value of IO permits reading, writing, overwriting, and purging.</li> <li>• A value of IN permits reading but not writing, overwriting, or purging.</li> <li>• A value of OUT permits writing, overwriting, or purging, but not reading.</li> </ul>
SENSITIVEDATA	When this file attribute is set to TRUE, it causes the disk or pack areas assigned for a file to be overwritten with an arbitrary pattern before the disk space is returned to the system for reallocation.
SINGLEUNIT	Indicates whether or not areas for a disk file are to be allocated from a single family member. The default value is FALSE.



Table 3-4 identifies the FTP transform attributes that can be specified in the <copy file> construct.

**Table 3-4. FTP Transform Attributes for Copying Files**

FTP Transform Attribute	Meaning
FTPSTRUCTURE	<p>Identifies the structure of the file that is being transferred.</p> <p>FTPFILE represents the file structure where there is no internal structure. The file is considered to be a continuous sequence of data bytes. This is the default structure used when transferring a file if FTPSTRUCTURE is not specified.</p> <p>FTPRECORD represents a record structure where the file is made up of sequential records.</p>
FTPTYPE	<p>Identifies the type of code format (either ASCII, EBCDIC, or image) in which the characters of the file are represented.</p> <p>ASCII NONPRINT represents the file in ASCII format without vertical format information. Vertical format control characters provide printer instructions such as carriage return (CR), line feed (LF), new line (NL), form feed (FF), and so on. The ASCII format is the default code format type.</p> <p>EBCDIC NONPRINT represents the file in EBCDIC format without vertical format control information.</p> <p>IMAGE represents the file in 8-bit transfer bytes. Data is sent as contiguous bits and must be stored as contiguous bits by the receiving host. If the receiving host must store the data in a manner such that the file (or record for a record-structured file) necessitates the padding of the file with 0 (zeroes) to some convenient boundary such as byte, word, or block, then the zeroes must be placed at the end of the file or record and the padding bits must be identifiable.</p>

Table 3-5 identifies the FTAM transform attributes that can be specified in the <copy file> construct. These file attributes apply to those files that are transferred to a remote host when the FTAM transfer service is used. Because some computer systems have implemented only the minimal level of support as adopted by the National Institute of Standards and Technology (NIST), the options for the FTAM transform attributes are limited.

**Table 3-5. FTAM Transform Attributes for Copying Files**

FTAM Transform Attribute	Meaning
BLOCKSTRUCTURE	<p>The BLOCKSTRUCTURE attribute specifies the string significance of the FTAM file as Not Significant. The mnemonic value for BLOCKSTRUCTURE must be EXTERNAL. This value indicates that records will be variable-length records.</p>
CREATEPASSWORD	<p>Use the CREATEPASSWORD attribute to specify the password string that you want to assign to a newly created file. The password provides additional security information that some computer systems require before the new file can be created. This security information is in addition to the log-on security that is passed to the remote host through the USERCODE attribute. Refer to the <i>A Series File Attributes Programming Reference Manual</i> for more information about the USERCODE attribute.</p> <p>The syntax for the password is the same as for the log-on password, which is described earlier in the explanation of the COPY command.</p> <p>The CREATEPASSWORD attribute is valid only when creating a new file on a remote host; otherwise, it is ignored.</p>
DOCUMENTTYPE	<p>The DOCUMENTTYPE attribute specifies the data type of the contents of the file and the structuring information of the file. A file transferred through FTAM transfer service must first be converted to one of four FTAM document types: FTAM-1, FTAM-2, FTAM-3, or INTAP-1.</p> <p>Any A Series file can be transferred as an FTAM-3 document. If a remote host does not support FTAM-2 files, the FTAM-2 file is transferred as an FTAM-1 file. Some file characteristics are not retained during this process of simplification.</p>

continued

Table 3-5. FTAM Transform Attributes for Copying Files (cont.)

FTAM Transform Attribute	Meaning														
EXTMODE	<p>This attribute specifies the external or physical character encoding of the file records. The character encoding used for a file is designated by a universal class value. The universal class values allowed for the FTAM transform attribute EXTMODE are ISOGRAPHICSTRING, ISOVISIBLESTRING, IA5STRING, and OCTETSTRING.</p> <p>The following identifies the allowed combinations of the A Series source file's file attribute EXTMODE with the FTAM transform attribute EXTMODE.</p>														
	<table border="0"> <tr> <td style="text-align: right;">Source File EXTMODE</td> <td style="text-align: left;">FTAM Transform Attribute EXTMODE</td> </tr> <tr> <td style="text-align: right;">-----</td> <td style="text-align: left;">-----</td> </tr> <tr> <td style="text-align: right;">Any value acceptable to FTAM</td> <td style="text-align: left;">OCTETSTRING</td> </tr> <tr> <td style="text-align: right;">ASCII</td> <td style="text-align: left;">IA5STRING, ISOGRAPHICSTRING, ISOVISIBLESTRING</td> </tr> <tr> <td style="text-align: right;">EBCDIC</td> <td style="text-align: left;">IA5STRING, ISOGRAPHICSTRING, ISOVISIBLESTRING</td> </tr> <tr> <td style="text-align: right;">IA5STRING</td> <td style="text-align: left;">ISOGRAPHICSTRING</td> </tr> <tr> <td style="text-align: right;">ISOVISIBLESTRING</td> <td style="text-align: left;">IA5STRING</td> </tr> </table>	Source File EXTMODE	FTAM Transform Attribute EXTMODE	-----	-----	Any value acceptable to FTAM	OCTETSTRING	ASCII	IA5STRING, ISOGRAPHICSTRING, ISOVISIBLESTRING	EBCDIC	IA5STRING, ISOGRAPHICSTRING, ISOVISIBLESTRING	IA5STRING	ISOGRAPHICSTRING	ISOVISIBLESTRING	IA5STRING
Source File EXTMODE	FTAM Transform Attribute EXTMODE														
-----	-----														
Any value acceptable to FTAM	OCTETSTRING														
ASCII	IA5STRING, ISOGRAPHICSTRING, ISOVISIBLESTRING														
EBCDIC	IA5STRING, ISOGRAPHICSTRING, ISOVISIBLESTRING														
IA5STRING	ISOGRAPHICSTRING														
ISOVISIBLESTRING	IA5STRING														
	<p>Some combinations of the A Series source file's file attribute EXTMODE and the character set specified for the FTAM transform attribute EXTMODE can result in the file not being transferred. An error message is displayed indicating that a translation between character sets was not allowed and the file was not transferred.</p>														
	<p>For remote hosts that provide only the minimum level of support for the FTAM-2 document type, use the ISOGRAPHICSTRING universal class value to translate the file to be transferred. The ISOGRAPHICSTRING universal class value enables a file to be transferred, but some loss of data is possible. For example, if an A Series source file has an EXTMODE mnemonic of ASCII, EBCDIC, or IA5STRING, then specifying a transformation of the file to ISOGRAPHICSTRING might cause data to be lost because of the differences in the character sets; however, the file transfer is performed.</p>														

continued

Table 3-5. FTAM Transform Attributes for Copying Files (cont.)

FTAM Transform Attribute	Meaning
EXTMODE	<p>If ISOGRAPHICSTRING is requested, the following warning is displayed:</p> <p>WARNING: DATA MAY BE LOST BY TRANSLATING FROM &lt;file's INTMODE&gt; TO GRAPHICSTRING.</p> <p>OCTETSTRING is the only universal class value that can be specified for document types FTAM-3 and INTAP-1, and is excluded from the list of valid universal class values for FTAM-1 and FTAM-2 document types.</p> <p>The only allowed translation of a character set to another must result in a character set that supports the entire character set of the original document. The intent is to prevent the loss of data because of the translation to the new character set.</p>

If an FTAM transform attribute is specified for a copy *from* a remote host, the following warning is displayed and the FTAM transform attribute is ignored:

WARNING: FTAM ONLY SUPPORTS A <FTAM TRANSFORM ATTRIBUTE> FOR COPIES "TO" A REMOTE HOST.

## File Transferring Services

The COPY or ADD command initiates one of four services for transferring files between hosts:

- A Series Native File Transfer (NFT)
- Host Services File Transfer (HOSTSERVICES or HS)
- File Transfer Protocol (FTP)
- File Transfer, Access, and Management (FTAM)

The selected file transfer service depends on the following circumstances:

- Whether a specific file transfer was specified in the COPY command
- Whether the involved source and destination hosts support the transfer service

The A Series Native File Transfer (NFT) enables you to copy files from one A Series host to another. Host Services enables you to copy files from one BNA host to another. If BNA is unavailable and both the source and destination are A Series hosts, then Host Services can use the Open Systems Interconnection (OSI) network to transfer files. The File Transfer Protocol (FTP) (used by Transmission Control Protocol and Internet Protocol (TCP/IP) Application Services) enables you to transfer files between A Series hosts and other hosts connected across a TCP/IP network. The File Transfer, Access, and Management (FTAM) service is part of the Open Systems Interconnection (OSI)

effort that enables you to transfer files between A Series hosts and other hosts that support FTAM.

Refer to the *A Series Distributed System Service (DSS) Operations Guide* for more information about the file transfer services. The features and requirements of these file transfer services when initiated by the COPY or ADD command are explained in the following paragraphs.

### **A Series Native File Transfer (NFT)**

NFT allows you to copy files between A Series systems, and provides all of the Host Services File Transfer features as well as the following additional features:

- File transfer resumption from the point of failure, so that already transferred data will not have to be transferred again
- Greater data transfer throughput than Host Services File Transfer
- The ability to transfer all disk files that Library Maintenance can copy
- Simultaneous transfer of files to several destinations
- Directory copies to and from remote hosts
- File transfers to and from Library Maintenance tapes
- Initiation of local Library Maintenance tasks at the local host and remote hosts
- Preservation of cataloged file generation information

Nondata files may be restricted from being copied through NFT if your usercode is not privileged or your security administrator has placed restrictions to preserve system integrity. A file is classified as a nondata file if its FILEKIND attribute has a value as a system file, code file, or compiler.

### **Host Services File Transfer**

Host Services allows you to copy files from one BNA or A Series OSI host to another. You can copy files between the local host and a remote host, or between two remote hosts. A directory can also be specified in the COPY command when transferring files from the local host to the remote host. The initiating host need not be either the source or the destination of the file.

Nondata files may be restricted from being copied through Host Services File Transfer if your usercode is not privileged or your security administrator has placed restrictions to preserve system integrity. A file is classified as a nondata file if its FILEKIND attribute has a value as a system file, backup file, code file, or compiler.

### **File Transfer Protocol (FTP)**

FTP allows you to transfer files across a TCP/IP network. The FTP user interface has many of the same characteristics as the Host Services File Transfer with the exception that all files using the FTP protocols are stored in a special FTP transfer format to

facilitate later transfer. The file kind of the transfer file is FTPDATA. If FTP transfers a file of file type IMAGE, the file kind is DATA. The FTP Utility allows you to convert an FTPDATA file to the A Series file format.

FTP transfers data files, source files, and code files; however, code files must be transferred with FTPTYPE = IMAGE. If the source file is compatible with a local compiler, you can compile the program to generate an operational code file.

Some additional limitations are imposed on outgoing files. A file with FILETYPE = 6 cannot be transferred. Only files with FILEORGANIZATION = NOTRESTRICTED can be transferred. See the *File Attributes Reference Manual* for details about the FILETYPE and FILEORGANIZATION file attributes.

The FTP transform attributes FTPSTRUCTURE and FTPTYPE can be used to specify the file structure and character code type, respectively, of the transferred file. Refer to "Specifying File Attributes," earlier in this section, for more information about the FTP transform attributes.

When IMAGE is specified for the FTP transform attribute FTPTYPE (FTPTYPE = IMAGE), the following attributes are applied to the file copied to an A Series system; however, an FTP header record is not placed at the beginning of the file:

```
FILESTRUCTURE = STREAM,  
ANYSIZEIO = TRUE,  
FRAMESIZE = 8,  
MAXRECSIZE = 1,  
FILEKIND = DATA,  
EXTMODE = OCTETSTRING,
```

Additionally, no translation or record size adjustment is made to the file. Each block of data is sent in the same format as it is read from the file. When a file is sent from an A Series host with the FTP transform attributes specified as FTPTYPE = IMAGE and FTPSTRUCTURE = FTPFILE, the carriage return, line feed (CRLF), and new line (NL) vertical format controls are not added to the end of each record.

### File Transfer, Access, and Management (FTAM)

FTAM is part of the Open Systems Interconnection (OSI) standard that provides file exchange and file management services. Transferring files between hosts that support FTAM is one of the services provided by FTAM.

Three classes of files can be transferred using FTAM. These file classes are referred to as *document types* to describe the characteristics of a file. The following are the supported document types:

Document Type	Characteristics
FTAM-1 (unstructured text file)	A file that consists of a single data unit (record) that contains character strings. The file must be accessed as a whole.
FTAM-2 (sequential text file)	A file that consists of data units (records), each of which contains character strings. The file can be accessed at the individual data unit level in a sequential mode.
FTAM-3 and INTAP-1 (unstructured binary file)	A file that consists of a single data unit (record) that contains binary strings. The file must be accessed as a whole.

The file attribute DOCUMENTTYPE is used to store the document type of a file. The files of the listed document types are stored on an A Series disk as character stream data files with a FILEORGANIZATION of NOTRESTRICTED.

An existing A Series file that does not have a specified DOCUMENTTYPE attribute is treated as one of the FTAM document types based on the FILEKIND and EXTMODE file attributes of that file. The following determines the FTAM document type assignment:

- A file with a FILEKIND value of SEQDATA, CSEQDATA, TEXTDATA, or <compiler>SYMBOL ( where <compiler>SYMBOL represents a kind of compiler symbol file, such as ALGOLSYMBOL or NDLSYMBOL) is considered a symbolic file. Symbolic files normally contain only displayable characters.
- A symbolic file with an EXTMODE value other than SINGLE or OCTETSTRING defaults to an FTAM-2 document.
- A file with a FILEKIND value not in the symbolic file list or with an EXTMODE value of either SINGLE or OCTETSTRING defaults to an FTAM-3 document. The FTAM-3 and INTAP-1 document types allow any binary value within an 8-bit frame.

## Copying Files between Hosts

When you initiate a COPY command, the transfer service selected is determined by the specified transfer service in the COPY command. If you do not specify a transfer service in the COPY command, the transfer service is selected for the following hosts:

- If the transfer is between A Series BNA hosts, either NFT or Host Services File Transfer is used.
- If the transfer is between non-A Series BNA hosts, Host Services File Transfer is used.
- If the transfer is between A Series OSI hosts, either Host Services File Transfer or FTAM is used.
- If the transfer is between hosts connected to a Transmission Control Protocol/Internet Protocol (TCP/IP) network, then the File Transfer Protocol (FTP) is used.
- If the transfer is between OSI hosts, FTAM is used.

The following situations also affect which file transfer service is chosen when a transfer service is not specified in the COPY command:

- Nonconcurrent file transferring between hosts  
NFT is selected if both the local host and the remote host can support NFT. If one host cannot support NFT, Host Services File Transfer, FTP, or FTAM is selected.
- Concurrent file transferring between several hosts  
The appropriate service for each pair of source and destination hosts is selected. First NFT is considered, then Host Services File Transfer, FTP, and finally FTAM.

If an initiating host participates in a file transfer between remote source and destination hosts (that is, the file is transferred through the initiating host), then all three hosts must use the same transfer service. FTAM transfer service does not support this file transfer method.

The following list shows the syntax construct necessary for the COPY command to specify a particular file transfer service:

File Transfer Service	COPY Command Syntax
NFT	Specify NFT for the <transfer service> construct.
Host Services File Transfer	Specify HOSTSERVICES (or its synonym, HS) for the <transfer service> construct.
FTP	Specify FTP for the <transfer service> construct.
FTAM	Specify FTAM for the <transfer service> construct.

If FTP is used to transfer files, any files copied reside on disk as FTPDATA files that are specially formatted. FTPDATA is not a standard A Series file kind and must be converted before it can be processed, printed, or viewed by system software. The FTP utility program can convert an FTPDATA file to a conventional A Series format with the title and certain other attribute values that you designate. For more information on converting FTPDATA files with the FTP utility, refer to the *DSS Operations Guide*.

## Restrictions on Copying Files between Hosts

When you use the COPY command to copy files, each of the file transfer services have certain restrictions that must be observed. These restrictions are explained in the following paragraphs.

### NFT File Transfer Restrictions

The following restrictions apply when you are transferring files through NFT:

- If you specify the CATALOG option, the cataloged indicator in the disk file header of the destination file is turned on, indicating that the file is cataloged, but a backup entry referencing the source file is not placed in the catalog of the destination host.
- Do not specify the BACKUP option.



- Do not specify the ONTO option of the <copy file> construct.
- Do not use tapes as destinations, unless you specify a single source volume.
- Do not use tapes as sources, unless you specify a single destination volume.
- Do not specify the INTERCHANGE or USERCODE attributes in the <source volume attribute list> or the <destination volume attribute list>.
- Interrupted file transfers to disk where a file title contains all 13 identifiers can only be restarted from the beginning of the file.
- Interrupted file transfers to tape can only be restarted from the beginning of the file.

### Host Services File Transfer Restrictions

The following restrictions apply when you transfer a file through Host Services File Transfer:

- Do not specify a directory unless it resides on the local initiating host.
- Do not specify the COMPARE, CATALOG, or BACKUP options.
- Do not use the ONTO option of the <copy file> construct.
- Copy from disk to disk only.
- Specify only the attributes KIND and HOSTNAME in the source volume attribute list and the destination volume attribute list.

### FTP File Transfer Restrictions

The following restrictions apply when you transfer a file through FTP:

- Do not specify a directory.
- Do not specify the COMPARE, CATALOG, or BACKUP options.
- Do not use the ONTO option of the <copy file> construct.
- Copy from disk to disk only.
- Specify only the KIND, HOSTNAME, and USERCODE attributes in the source volume attribute list and the destination volume attribute list.
- Use only the volume name DISK in the source volume and destination volume constructs.
- Depending on the FTP implementation of a receiving non-A Series host, FTPSTRUCTURE = FTPFILE and FTPTYPE = ASCII NONPRINT might be the only file structure and character code type recognized. Other options for FTPSTRUCTURE and FTPTYPE transform attributes might result in the non-A Series host returning an error message indicating that the specified FTPSTRUCTURE or FTPTYPE is not allowed at the remote host.
- FTP does not distinguish between the ADD and the COPY commands. Any file that exists on the host under the name specified in the ADD command is overwritten.

## FTAM File Transfer Restrictions

The following restrictions apply when you transfer a file through FTAM:

- Do not specify a directory.
- Do not specify the COMPARE, CATALOG, VERIFY, or BACKUP options.
- Do not use the ONTO option of the <copy file> variable.
- Copy only from disk to disk.
- In the source volume and destination volume attribute lists, specify only the KIND, HOSTNAME, and USERCODE attributes.
- If the destination host already has a file by the name specified in the COPY command, that file is replaced with the transferred file. If the ADD command is used, the transfer is not accepted by the remote host.

## Restarting Interrupted File Transfers

When a host or network failure interrupts a file transfer request initiated from CANDE, reenter the file transfer request to resume transfer. A WFL job containing the COPY command automatically restarts if the failure was caused by a halt/load or if the job was written to restart after an unsuccessful file transfer. For more detailed information about WFL jobs that restart, refer to the *WFL Reference Manual*.

The results of restarting the COPY command vary based on the type of file transfer:

- If the file transfer is local, all files named in the COPY command are transferred again when the job restarts.
- If the file transfer is remote and a transfer service other than NFT is used, all files named in the COPY command are transferred again when the job restarts.
- If the file transfer is remote and the transfer service is NFT, the FROMSTART option determines the effect of restarting the copy.
  - If FROMSTART is not specified, NFT automatically resumes the COPY command so that already transferred data is not transferred again.

The resumption point depends on the kind of destination volume and the characteristics of the files being transferred. For more information about file transfer resumption in NFT, refer to the *DSS Operations Guide*.

- If FROMSTART is specified, all files named in the COPY command will be completely transferred again, regardless of whether NFT file transfer resumption could have been used.

**Note:** When FROMSTART is specified, any previously created NFTTEMP recovery files matching the COPY request are removed.

## Examples

The following five examples show various aspects of using the COPY command.

### Example 1

The following examples show variations of the COPY syntax:

- This statement copies the file X from DISK to disk Y:

```
COPY X TO Y(KIND=DISK);
```

- This statement copies the file X from tape T to DISK:

```
COPY X FROM T(KIND=TAPE);
```

- This statement copies the file X/Y and all files under the directory Z/= from disk P to both tapes T1 and T2:

```
COPY X/Y, Z/= FROM P(KIND=DISK) TO T1(KIND=TAPE), TO T2(KIND=TAPE);
```

- This statement copies the file X from DISK to DISK, changing the name of the new file to Y:

```
COPY X AS Y;
```

- This statement copies the file X from DISK (at local host) to DISK at remote host HOSTB, changing the name of the new file to Y:

```
COPY X AS Y FROM DISK TO DISK(HOSTNAME=HOSTB);
```

- This statement copies the file WEEKLY\_SUMMARY (from DISK at local host) to DISK at remote host HOSTB, renaming the file as WEEKLY/SUMMARY:

```
COPY WEEKLY_SUMMARY AS WEEKLY/SUMMARY TO DISK(HOSTNAME=HOSTB);
```

- This statement copies the file TIME/SUM from the disk ENGDATA at the host HOSTE as the file ENG/TIME/SUM on the disk ACCTDATA at the host HOSTA:

```
COPY TIME/SUM AS ENG/TIME/SUM FROM ENGDATA(KIND=DISK,HOSTNAME=HOSTE)
TO ACCTDATA(KIND=DISK,HOSTNAME=HOSTA);
```

- This statement copies all files in the directory SYMBOL/= from the tape REL to DISK, and all files in the directory SYSTEM/= from the tape REL to PACK:

```
COPY SYMBOL/= FROM REL(KIND=TAPE) TO DISK,
SYSTEM/= FROM REL(KIND=TAPE) TO PACK;
```

- This statement copies all files from DISK to the tape T, adding X/= as a prefix to each file name. It also copies the file A (without changing its name) from disk B to tape T.

```
COPY = AS X/= FROM DISK, A FROM B(KIND=DISK) TO T(KIND=TAPE);
```

- This statement copies and compares all files from the tape T to disk D. Immediately after each file is copied to disk D, it is compared with the original file on tape T.

## COPY Command (cont.)

---

```
COPY & COMPARE = FROM T(KIND=TAPE) TO D(KIND=DISK);
```

- When run from a privileged usercode, this statement copies all files from tape T to disk D, including files with and without usercodes:

```
COPY *= FROM T(KIND=TAPE) TO D(KIND=DISK);
```

- This statement copies the file X from tape T onto (on top of) file Y on DISK. The files must have the same blocking, row size, and number of rows. If the files are incompatible, a run-time error occurs.

```
COPY X ONTO Y FROM T(KIND=TAPE);
```

- This statement copies the file X from DISK to the tape T and compares, bit for bit, file X on DISK and file X on tape T. An entry is placed in the system catalog indicating that tape T contains a backup copy of file X.

```
COPY & COMPARE & BACKUP X TO T(KIND=TAPE);
```

### Example 2

The following examples illustrate the COPY syntax used when NFT is used to transfer files between A Series hosts:

- This statement copies the file A from a tape named ACCTS on the local host to a pack named PACK on remote host HOSTB:

```
COPY [NFT] A FROM ACCTS TO PACK(HOSTNAME=HOSTB);
```

- This statement copies the file A to a pack named PACK on host HOSTB. For this example, assume that the COPY command is part of a WFL job that is restarted if the file transfer fails. If the WFL job restarts, file A is completely retransferred because FROMSTART is specified in the COPY command.

```
COPY [NFT, FROMSTART] A TO PACK(HOSTNAME=HOSTB);
```

### Example 3

The following examples illustrate the COPY syntax used when Host Services File Transfer is used to transfer files across a BNA network:

- This statement copies the file X from DISK (at local host) to DISK at remote host HOSTB, changing the name of the new file to Y. The HOSTSERVICES option is used to designate that Host Services File Transfer should be used to transfer the file.

```
COPY [HOSTSERVICES] X AS Y FROM DISK TO DISK(HOSTNAME=HOSTB);
```

- This statement copies the file WEEKLY\_SUMMARY (from DISK at local host) to DISK at remote host HOSTB, renaming the file as WEEKLY/SUMMARY so it can be transferred. In this example, HOSTB does not support NFT, so Host Services File Transfer is selected.

```
COPY WEEKLY_SUMMARY AS WEEKLY/SUMMARY TO DISK(HOSTNAME=HOSTB);
```

- This statement copies the file TIME/SUM from the disk ENGDATA at the host HOSTE as the file ENG/TIME/SUM on the disk ACCTDATA at the host HOSTA. In this example, HOSTE supports NFT but HOSTA does not. Host Services File Transfer is selected to copy the files.

```
COPY TIME/SUM AS ENG/TIME/SUM FROM ENGDATA(KIND=DISK,HOSTNAME=HOSTE)
TO ACCTDATA(KIND=DISK,HOSTNAME=HOSTA);
```

#### Example 4

The following examples illustrate the COPY syntax used when FTP transfers files across a TCP/IP network:

- This statement copies the file PAYROLL/EXEMPT from DISK at the local host to DISK at the remote host HQSYS1, changing the name of the new file to PAYROLL/SALARIED. The FTP option is used to designate that FTP File Transfer should be used to transfer the file.

```
COPY [FTP] PAYROLL/EXEMPT AS PAYROLL/SALARIED FROM DISK TO
DISK(KIND=PACK, HOSTNAME=HQSYS1, USERCODE=RMPAYR/MONEY/735);
```

- This statement copies the file PAYROLL/TIMECARDS/061490 from DISK at the local host as the file [PAY735]TC061490.DAT to DISK at the remote host HQSYS1. Log-on information is passed to the remote host HQSYS1 with the USERCODE attribute.

```
COPY PAYROLL/TIMECARDS/061490 AS '[PAY735]TC061490.DAT'
TO DISK(HOSTNAME=HQSYS1, USERCODE=RMPAYR/MONEY/735);
```

- This statement copies the file [PAY735]90AWARDS.DAT from DISK at the remote host HQSYS1 as the file PAYROLL/AWARDS/1990 to DISK at the local host:

```
COPY '[PAY735]90AWARDS.DAT' AS PAYROLL/AWARDS/1990 FROM
DISK(HOSTNAME=HQSYS1, USERCODE=RMPAYR/MONEY/735) TO DISK;
```

- This statement copies the file [PAYGEN]NEWFORMS.DAT from DISK at the remote host HQSYS1 as the file PAYROLL/NEWFORMS to DISK at the other remote host RMSYS4. Log-on information is passed to the remote hosts with the USERCODE attribute.

```
COPY '[PAYGEN]NEWFORMS.DAT' AS PAYROLL/NEWFORMS
FROM DISK(KIND=PACK, HOSTNAME=HQSYS1, USERCODE=RMPAYR/MONEY/735)
TO DISK(KIND=PACK, HOSTNAME=RMSYS4, USERCODE=PAY735/CASH/735);
```

- This statement copies the file PAYROLL/TIMECARDS/070690 from DISK at the local host as the file [PAY832]TC070690.DAT on DISK at the remote host HQSYS1. The FTP transform attribute FTPTYPE identifies the resultant code type of the file that is to be transferred.

```
COPY PAYROLL/TIMECARDS/070690 AS '[PAY832]TC070690.DAT'
(FTPTYPE=IMAGE) FROM DISK TO
DISK(HOSTNAME=HQSYS1, USERCODE=RMPAYR/MONEY/832);
```

**Example 5**

The following example illustrates the COPY syntax when FTAM is used to transfer a file between an A Series host and a non-A Series host.

- This statement copies the file FILEZ from MYFAMILY at the local A Series host as the file C:\FILEZ.DOC to DISK at the remote non-A Series host OSIHOST. The FTAM option is used to designate that the FTAM file transfer should be used to transfer the file. Log-on information is passed to the remote host with the USERCODE attribute.

```
COPY [FTAM] FILEZ AS 'c:\filez.doc' FROM MYFAMILY(PACK) TO  
DISK(HOSTNAME = OSIHOST, USERCODE = TOM/123)
```

- This statement copies the file C:\FILEZ.DOC from DISK at the remote non-A Series host as FILEZ to MYFAMILY at the local A Series host.

```
COPY 'c:\filez.doc' AS FILEZ FROM DISK(HOSTNAME = OSIHOST, USERCODE =  
TOM/123) TO MYFAMILY(PACK)
```

- This statement copies the file FILEZ from MYFAMILY at the local A Series host as the file 'a.file' to DISK at the remote non-A Series HOST. The FTAM transform attribute DOCUMENTTYPE is specified to designate the document type of the file on the remote host.

```
COPY [FTAM] FILEZ AS 'a.file' FROM MYFAMILY(PACK)  
(BLOCKSTRUCTURE = EXTERNAL, DOCUMENTTYPE = FTAM2,  
EXTMODE = ISOGENERALSTRING) TO DISK(HOSTNAME=OSIHOST, USERCODE=TOM/123)
```

**Example 6**

The following examples illustrate the ADD syntax:

- This statement copies the file X/Y from tape T to DISK, only if no file named X/Y already resides on DISK:

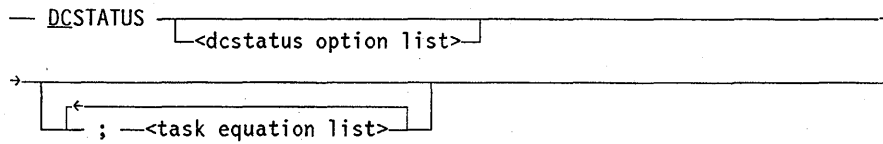
```
ADD X/Y FROM T(KIND=TAPE);
```

- This statement copies files in the directory Z/= from tape T to disk R and to DISK. Any files already resident on the destination volumes are not copied. Note that different files might be copied to R and DISK, depending on what is already resident on each destination volume before the ADD is done.

```
ADD Z/= FROM T(KIND=TAPE) TO R(KIND=DISK), TO DISK;
```

## DCSTATUS

### Syntax



### Explanation

The DCSTATUS command causes execution of the DCSTATUS utility. DCSTATUS allows a run-time analysis of the current state of the data comm subsystem to be performed. The <dcstatus option list> construct must consist of a string of standard options allowed by DCSTATUS. If the <dcstatus option list> is not specified, the default is the STATION option with <LSN> set to the user's LSN.

All output from DCSTATUS is directed to the user's terminal unless the file LINE is file-equated to KIND=PRINTER in the input statement; the ALL, NSP, and CLUSTER/LSP options produce voluminous output. Refer to the *A Series System Software Support Reference Manual* for more information about DCSTATUS and the DCSTATUS options.



**Examples**

DC  
#RUNNING 3854

STATION 220

SYSTEM/DCSTATUS (3.7.200) DATE : 06/10/87 TIME : 09:49:55  
DCPREFIX: LFA15/CD  
STATION 220

DCC STATION TABLE

ENABLED : READY : ATTACHED :

MCS = 2: LSN = 220 : PSEUDOSTATION : WIDTH = 80

STATION REMOTE TYPE = 0: RETRY COUNT = 0: NIF INDEX = 219

PRIMARY Q = 14,CURRENT Q = 14,STN Q = 0,PSEUDOMCS = 1

TRANSFERRED : DLS : UNASSIGNED ATTACHED TO FILE 22 REL STN NO = 1

NORMAL TERMINATION

#

DCSTATUS GRAPH;FILE LINE(KIND=PRINTER)

#RUNNING 40000

#

## DELETE Command

---

# DELETE

### Syntax

— DELETE —<sequence range list>|  
          | ALL

### Explanation

The DELETE command causes the lines specified in the <sequence range list> construct to be deleted from the work file. If DELETE ALL is entered, the contents of the work file are deleted, but the title and other attributes are preserved. If the work file is unnamed, it is removed by the DELETE ALL form.

### Examples

```
L  
1000 LINE1  
2000 LINE2  
3000 LINE3  
4000 LINE4  
5000 LINE5  
6000 LINE6  
#
```

```
DELETE 1000  
#
```

```
L  
2000 LINE2  
3000 LINE3  
4000 LINE4  
5000 LINE5  
6000 LINE6  
#
```

```
DELETE 2500-3000,5000-END  
#
```

```
L  
2000 LINE2  
4000 LINE4  
#
```

## DESTNAME

### Syntax

```

— DESTNAME —————|
|                                     |
| = ———— <station name> ———— |
| * ———— |
| . ———— |

```

### Explanation

The DESTNAME command provides a means of specifying a non-CANDE station for the printed output of a task. This command interrogates the current DESTNAME and changes the current DESTNAME. The DESTNAME is supplied on all task executions explicitly initiated by the user and on CANDE-initiated tasks, such as CANDE WRITE. The current DESTNAME may be overridden by an explicit specification on a RUN or COMPILE request.

The default DESTNAME for a particular user is determined by the CANDEDESTNAME for that user in the USERDATAFILE. To change the default DESTNAME in the USERDATAFILE, MAKEUSER must be invoked by the installation. (Refer to the *A Series Security Administration Guide* for further information about MAKEUSER.) The default DESTNAME for a given usercode is set at the beginning of each session associated with that usercode. The CANDE DESTNAME command can be used to override the default DESTNAME during a session.

When DESTNAME is the only input entered, CANDE reports the current value of the DESTNAME.

If the station name is supplied, it is saved by CANDE as the current value of DESTNAME until another DESTNAME action takes place. The station name can have a maximum of 107 characters to specify the DESTNAME.

The asterisk (\*) option causes the current value of the DESTNAME to return to the default of the user associated with the session.

The period (.) option causes the DESTNAME to be set to null. In this case, DESTNAME is not supplied by CANDE on any task execution request.

If the DESTNAME is explicitly changed during a session, backup requests (such as BACKUPPROCESS) search for BD, BP, REMLP <nn>, and REMCP <nn> files, where <nn> is the NDII-defined MCS number for each destination station used during that session. Searches are made on DISK, PACK, and the DL BACKUP device that was in effect at the beginning of the session.

When the value of DESTNAME is explicitly changed in a RUN or EXECUTE command, the new value overrides the former value for the session for that request only. No abbreviation for DESTNAME is accepted in a RUN or EXECUTE command. (Refer to the RUN and EXECUTE statements for the correct syntax.) Changing the DESTNAME for execution of any given task in the session does not affect the current DESTNAME for the session.

## DESTNAME Command (cont.)

---

If the PRINTDEFAULTS attribute DESTINATION is specified for the user's session, then the station name specified by DESTNAME is overridden by the PRINTDEFAULTS DESTINATION value. The following message is displayed when either DESTNAME or PRINTDEFAULTS DESTINATION is specified or changed and the other is already specified:

WARNING: PRINTDEFAULTS DESTINATION OVERRIDES DESTNAME.

This warning is also displayed at CANDE log-on time if both CANDEDESTNAME and PRINTDEFAULTS are specified for the user in the USERDATAFILE. See the *Security Administration Guide* for information about the USERDATAFILE and its file attributes.

### Examples

DESTNAME.

DEST RJE

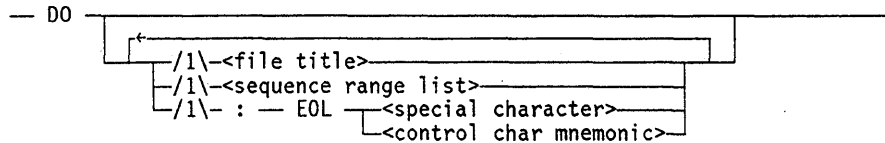
DEST\*

DESTNAME



# DO

### Syntax



### Explanation

The DO command causes a file (or portions thereof) of CANDE input to be displayed and performed synchronously in the session. Valid input includes CANDE commands (except control commands) and single-line insertions.

The specified input is entered in the user's queue and performed on completion of the DO command. This queued input is treated like any other queued input except that each command is displayed when it is removed from the user's queue. When this queue is displayed with the ?SHOW control command, an entry from a DO command is preceded by a *D*.

Most commands (such as CHANGE, SECURITY, and TITLE) attempting to act on a file that is not present cause the next command in the DO file to be queued and the message "#QUEUED INPUT PENDING" to be displayed at the originating station. The REMOVE command is an exception to this situation because the fact that a file is not present is interpreted as the successful removal of the file.

If no < file title > is specified, the user's work file is used. If no < sequence range list > is specified, the entire file is used. Only the text field of the file is significant, and trailing blanks are removed. If trailing blanks are significant, an end-of-line character may be specified (via the :EOL syntax) to allow specification of significant characters.

If the :EOL < special character > clause is specified, that character may be used to mark the end of an input line by placing it after the last column of valid input on the line. All further information on the input line is ignored; that is, the EOL character effectively ends the line. This allows comments to be placed on the line. It can also be helpful when trailing blanks are to be included as valid input (for example, in a FIX command).

CANDE can also invoke the DO command automatically when a user logs on. Refer to the discussion on startup files in Section 1, "General Information," for additional information.

**Examples**

```
LIST X
#FILE (UZER)X ON USERPACK
100 WHAT;REM;REM Y
200 MAKE Y ALGOL
300 100TEXT FOR Y
400 200LIST Y
500 300MORE TEXT FOR Y
600
700 SAVE;L
800 DO 200
900 WHAT
#

DO X
#FILE (UZER)X ON USERPACK
#
WHAT;REM;REM Y
#WORKFILE TEMP: SEQ, 5 RECORDS (THRU 5000), SAVED
#
# (UZER)Y ON USERPACK REMOVED
MAKE Y ALGOL
#WORKFILE Y: ALGOL
100TEXT FOR Y
200LIST Y
300MORE TEXT FOR Y
#
SAVE;L
#UPDATING
#WORKSOURCE Y SAVED
100 TEXT FOR Y
200 LIST Y
300 MORE TEXT FOR Y
#
DO 200
#WORKFILE Y
#
LIST Y
#FILE (UZER)Y ON USERPACK
100 TEXT FOR Y
200 LIST Y
300 MORE TEXT FOR Y
#
WHAT
#WORKFILE Y: ALGOL, 3 RECORDS (THRU 300), SAVED
```

## DO Command (cont.)

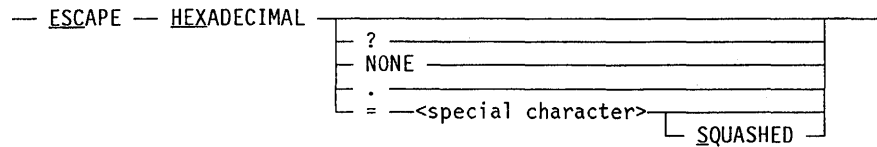
---

```
LIST SHORT
#FILE (UZER) SHORT ON USERPACK
100 RUN MY/UTILITY;VALUE=23 % USES NEW OPTION GROUP
200 START MY/JOB ("FOO TOO") % DOES BOTH
#
DO SHORT:EOL %
```



# ESCAPE

## Syntax



## Explanation

The *ESCAPE* command sets or resets the hexadecimal escape character. The assigned character can be a special character that is used to delimit a sequence of EBCDIC hexadecimal values for nonprintable characters. Any EBCDIC internal code that does not represent an alphanumeric character, a special character, or a blank represents a nonprintable character. Refer to Section 2, “Basic Constructs,” for the list of valid special characters.

Because nonprintable characters cannot be directly entered or displayed in a file that is being edited, they are represented by the corresponding EBCDIC hexadecimal values. When entering nonprintable characters into a file, the EBCDIC hexadecimal values must be delimited with the hexadecimal escape character. The EBCDIC hexadecimal values of those nonprintable characters are displayed with the hexadecimal escape character delimiters when the file is listed.

While the hexadecimal escape character is set, each nonprintable character is displayed in its corresponding 2-digit EBCDIC hexadecimal value, with single blank spaces before and after it. The value, including the spaces, is delimited by the hexadecimal escape character. If two or more nonprintable characters appear as a string—that is, with no printable characters between them—the entire string is delimited by the hexadecimal escape character; single blank spaces appear between each hexadecimal value, as well as before and after the string. Printable characters appear as they are entered.

For example, suppose you enter into a file the text string *ABC*, followed by the nonprintable characters end-of-text (ETX) and line feed (LF), followed by the text string *def*, and the hexadecimal escape character is the at sign (@); the resulting series of text and nonprintable characters is displayed as follows:

```
ABC@ 03 25 @def
```

The ETX and LF characters are represented by the EBCDIC hexadecimal values 03 and 25, respectively, and the at sign (@) delimits these nonprintable characters.

ESCAPE HEXADECIMAL can be abbreviated as ESC HEX.

ESCAPE HEX  
ESCAPE HEX ?

When the *ESCAPE HEX* or *ESCAPE HEX ?* form of the command is entered, CANDE displays the currently set hexadecimal escape character. If an escape character is not set, then the following message is displayed:

## ESCAPE Command (cont.)

---

#NO ESCAPE HEXADECIMAL CHARACTER DEFINED

**ESCAPE HEX NONE**  
**ESCAPE HEX .**

When the *ESCAPE HEX NONE* or *ESCAPE HEX .* form of the command is entered, the hexadecimal escape character is reset. Note that you must define a hexadecimal escape character if you want to insert nonprintable characters into a file and display their EBCDIC hexadecimal values.

If a hexadecimal escape character is not defined and a file contains nonprintable characters, then those nonprintable characters are sent to the terminal when the file is displayed. Unisys recommends that you define a hexadecimal escape character because terminal communication can be adversely affected if the hexadecimal escape character is not set and the system attempts to display a file containing data comm control characters.

**ESCAPE HEX = <special character>**

When the *ESCAPE HEX = <special character>* form of the command is entered, the character indicated is set as the escape character for hexadecimal values. Note that the semicolon (;) cannot be set as a hexadecimal escape character.

To prevent confusion, avoid setting the hexadecimal escape character to a character that is used frequently in a file and is subject to display. If a character is used in a file and that character is also defined as the hexadecimal escape character, then you must enter that character twice for each time it should occur in the file.

For instance, if the dollar sign (\$) is defined as the hexadecimal escape character, and the dollar sign must be entered in the file, then two dollar signs must be entered and both dollar signs are displayed. Suppose you want to enter the following line in a file:

A compiler control record begins with a \$ in column one.

When the dollar sign is set as the hexadecimal escape character, you must enter the line as follows. Both dollar signs are also displayed.

A compiler control record begins with a \$\$ in column one.

**ESCAPE HEX = <special character> SQUASHED**

The *ESCAPE HEX = <special character> SQUASHED* form of the command strips the blank spaces from between the hexadecimal values of the nonprintable characters. This option is useful for displaying a file in CANDE page mode because a line with hexadecimal values is not wider when it is displayed than it is when it is entered. The displayed line is therefore less likely to exceed the terminal line width and be truncated.

The SQUASHED option is reset whenever the hexadecimal escape character is reset or whenever the hexadecimal escape character is set without the use of the SQUASHED option.

**Examples**

ESCAPE HEX  
#NO ESCAPE HEXADECIMAL CHARACTER DEFINED

ESCAPE HEX = @  
#ESCAPE HEXADECIMAL = @

ESCAPE HEX ?  
#ESCAPE HEXADECIMAL = @

MAKE TEMP  
#WORKFILE TEMP: SEQ

100SAMPLE TEXT @0074@ SOME MORE TEXT  
200 @ 03 @

LIST  
100 SAMPLE TEXT @ 00 74 @ SOME MORE TEXT  
200 @ 03 @  
#

300ENTER AT SIGN @@

LIST  
100 SAMPLE TEXT @ 00 74 @ SOME MORE TEXT  
200 @ 30 @  
300 ENTER AT SIGN @@  
#

ESCAPE HEX = @ SQUASHED

LIST TEMP  
100 SAMPLE TEXT @0074@ SOME MORE TEXT  
200 @30@  
300 ENTER AT SIGN @@  
#

ESCAPE HEX NONE  
#NO ESCAPE HEXADECIMAL CHARACTER DEFINED



## EXCLUDE

### Syntax

```
— EXCLUDE <file title> <sequence range list>
          <sequence range list> <file title>
```

### Explanation

The EXCLUDE command causes any line in the work file with the same sequence number as a line in the specified <file title> to be deleted. Unmatched lines in the excluded file are ignored.

The UPDATE option is implicitly invoked after the EXCLUDE option has been executed.

The EXCLUDE command is particularly helpful when used in combination with the FIND command FILE option. This command should not be used with type DATA or CDATA files because these files do not have sequence numbers stored in the text.

### Example

```
G TESTFILE
#WORKFILE TESTFILE: SEQ, 8 RECORDS, SAVED
```

```
LIST
100 A
200 *B
300 C
400 D
500 *E
600 *F
800 *T
#
```

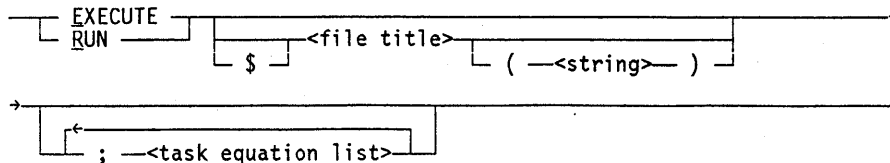
```
FIND*/:FILE EXCLUDE/FILE
#WORKFILE TESTFILE
#
```

```
EXCLUDE EXCLUDE/FILE
#UPDATING
#
```

```
LIST
100 A
300 C
400 D
#
```

## EXECUTE or RUN

### Syntax



### Explanation

The EXECUTE command causes execution of an object program. RUN is a synonym for EXECUTE.

The object work file is assumed if no file title is specified. An attempt to execute the work file when it has no object file causes the work file to be compiled and then executed. An attempt to execute a file title that does not have an associated object file results in an error message; the program is not automatically compiled. A dollar sign (\$) is specified to indicate that the given file title is not in the object file directory but is referenced as a normal user file. If the file title starts with an asterisk (\*) instead of a usercode, CANDE executes the file title instead of the associated object file. For example, RUN \*MYFILE is equivalent to RUN \$\*MYFILE. The file title must refer to a file accessible by the user.

In an EXECUTE command with file title specified, a single parameter may be optionally provided. The parameter must be a string whose length is limited only by the input record size of the terminal. The corresponding actual parameter must be a real array.

If the file to be executed is the work file, an UPDATE is done prior to the EXECUTE.

For more information about the modifier construct, refer to the modifier syntax in Section 2, "Basic Constructs." If modifier is used with the EXECUTE command, the attribute specified by the modifier applies to the executed task.

Input of more than one line of text is often desired when the EXECUTE command is used. More than one line of text can be entered by using the default (or *tasking only*) continuation character (%), or the current continuation character. (Refer to the CONTINUE command for details.)

If CANDE attempts to execute a code file that is not compatible with the machine, the following message is generated:

```
#DSED CODEFILE INCOMPATIBLE WITH THIS MACHINE
```

This message occurs, for example, in response to a RUN X statement when OBJECT/X was compiled with TARGET = A9 but the run was attempted on a B 6900.

**Examples**

EXECUTE  
#RUNNING 3437

E \$CODEFILE("WITH A STRING PARAMETER");FILE DCOM(KIND=REMOTE)  
#RUNNING 3441

R UTILITY/LOADER ON MYPACK ("USING THIS STRING")  
#RUNNING 3445

EX (ANYUSER)ANYPROGRAM ON HISPACK ("USING ANYSTRING")  
#RUNNING 3448

RUN \*SYSTEM/DUMPALL ON PACK ("TEACH");FILE LINE(KIND=REMOTE)  
#RUNNING 3449





The target family, substitute family, and alternate family are all family names.

The <family specification> construct is used to redirect references to files on the target family. Requests for an existing file on the target family search the substitute family and then the alternate family (if it exists). New files that request the target family are created on the substitute family.

When creating or referencing files through CANDE, the files are always associated with the family DISK. Family substitution is effective for CANDE functions only if the target family is DISK. Thus, a family specification of PACK=X ONLY does not affect files referenced by CANDE but affects tasks that reference files on the family PACK. If the target family is not DISK, then CANDE creates all files on the family DISK.

The word TAPE is an invalid family name.

### **Examples**

```
FAM DISK=PACK ONLY
#FAMILY DISK = PACK ONLY
```

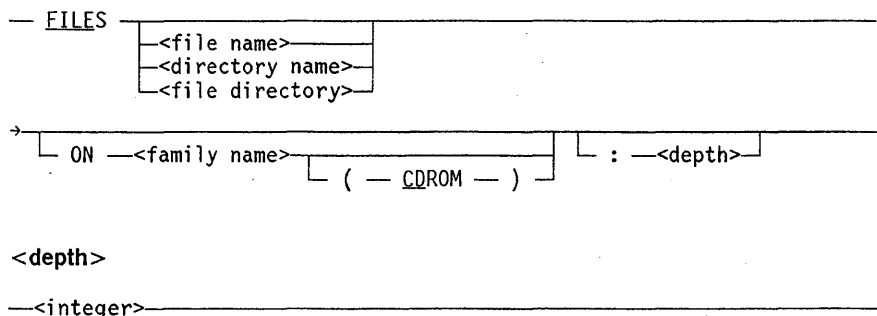
```
FAMILY.
#FAMILY.
```

```
FAM*
#FAMILY DISK = USERPACK OTHERWISE DISK
```

```
FAMILY X=Y OTHERWISE Z
#FAMILY X = Y OTHERWISE Z
```

## FILES

### Syntax



### Explanation

The FILES command causes CANDE to list the names and types of files in a user's library.

If FILES is entered, all files and directories in the user's library on the family name of the current work file are listed.

If a < file directory > is specified, the names and types of files under the directory for the current family names are listed.

If a < file name > is given, the name and type of the file on the current family name are listed. If the name supplied is both a file name and a directory name, all files under the directory name and the file name on the current family name are listed.

If a < directory name > is specified, the file names subordinate to that directory are listed. The equal sign (=) can be used to list all files under the directory. The following is an example of using the FILES command to list the files under a directory name:

```
FILES (LEDGER)PAYROLL/=
```

Note that if ON < family name > is specified, family substitution does not apply. If ON < family name > is followed by CDROM, the family name refers to a CD-ROM reader rather than a pack or disk. A CD-ROM reader is a read-only device that reads files stored on a compact disc.

The < depth > option can be used to indicate the level of retrieval of names to be listed.

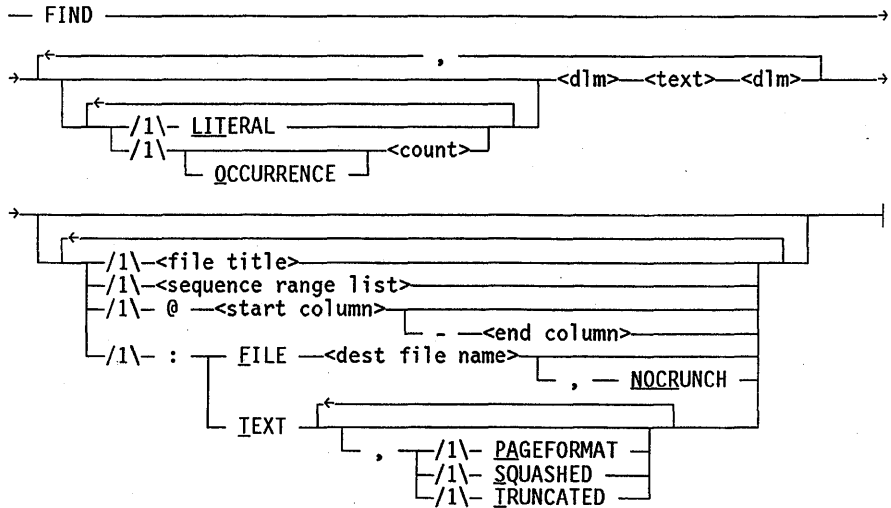
**Examples**

```
FILES
(AVERAGEUSER) ON DISK
. AFILE
. . A1 : SEQDATA
. . A2 : SEQDATA
. BFILE
. . B1 : SEQDATA
. . B2 : SEQDATA
. CFILE : SEQDATA
#
```

```
files : 1
(AVERAGEUSER) ON DISK
. AFILE
. BFILE
. CFILE : SEQDATA
#
```

# FIND

## Syntax



**<count>**

—<integer>

**<dest file name>**

—<file name>

## Explanation

The FIND command searches a file (by default, the work file) for appearances of specified target text. The line numbers of lines containing the target text are displayed at the user's terminal with an asterisk (\*) denoting line numbers where the target text appears more than once. The output option TEXT displays the lines of text in addition to the line numbers. The FILE output option directs the lines of text to a new file instead of the terminal.

The <text> construct specifies the target text and must be bounded by two matching delimiters, <dlm>. In a single FIND command, one or more target texts can be the subject of the search. The text can contain any characters, except the <dlm> character, and cannot be null.

If a hexadecimal escape character is set through the CANDE command ESCAPE, then the target text can contain hexadecimal values delimited by the defined hexadecimal escape character. However, the target delimiter (<dlm>) cannot be the same as the hexadecimal escape character. For more information about the hexadecimal escape character, refer to the ESCAPE command earlier in this section.

The value of <count> must be an integer. The <count> construct limits the search to the first <count> occurrences of the target. All occurrences after that number are

ignored. If more than one target text is specified, <count> applies only to its associated text.

When the OCCURRENCE option is used in conjunction with <count>, the FIND command locates and displays the *n*th occurrence of the target specified by the count. An integer value must be specified for the count whenever OCCURRENCE is used. If the target occurs more than once on a particular line, each occurrence on that line is counted. The integer value for the count cannot exceed 255.

The LITERAL option specifies that the mode of the search for the associated text is literal mode. The default, when LITERAL is not specified, is token mode. (A description of these search modes appears later in this explanation.)

The <file title> specifies the file to be searched and can be any file the user is allowed to access. When a file title is not specified, the work file is searched.

The <sequence range list> restricts the lines to be searched to one or more sequence ranges.

The at sign (@) option restricts the search to a column range on each line. The <start column> specifies the column number where the search is to begin. The <end column> specifies the column number where the search is to end. The column range can specify any part of the usable line except the sequence number field (for example, @81-90 is valid for type ALGOL files). The entire text field of the line is searched if no columns are specified. If only a start column is specified, the default end column is the last column of the text field. Thus, to search only column 1 of each line, the specification would be @1-1.

The TEXT option displays the entire line containing the target text, including the sequence number. When the TEXT option is used, all lines that have been marked with a current MARKID will have an asterisk between the sequence number and the contents of the line.

When the PAGEFORMAT option is specified, the output appears in a format similar to the page mode output. That is, the full length of the sequence number field is displayed, immediately followed by the contents of the text field (the blank or asterisk flag character is omitted).

For more information on the MARKID, refer to the GET command, the MAKE command, and the MARKID command.

The SQUASHED option causes a sequence of blanks to be printed as a single blank.

The TRUNCATED option causes the output line to be truncated to the character width of the terminal if necessary. By default, lines that exceed the terminal width are split across two or more lines.

The FILE option writes the entire line containing the target text to a new file in the user's library. The <dest file name> construct specifies the name of that new file. The new file is the same type as the work file or file title. Files created by the FIND command are crunched by default. If a crunched file is not desired, the output option

## FIND Command (cont.)

---

NOCRUNCH can be specified. If a file entitled < dest file name > already exists in the user's library, an error message is displayed.

The < dlm > character can be any special character except semicolon (;), comma (,), at sign (@), or colon (:).

CANDE uses two text searching modes. Both the FIND and the REPLACE commands use these searching modes. A token mode search is performed by default, and the literal mode search can be invoked by using the LITERAL option.

In literal mode, each line of the file is considered an arbitrary string of characters, including blanks. The search is successful whenever the sequence of characters in the target text exactly matches a sequence of characters in a line of the file. For example, if the target is *lit.a-b*. (periods are used as < dlm > characters), the following lines contain examples of matching target text:

```
x = a - b
x = aa - b
x = a - bb
```

However, the target *lit.a-b*. (periods are used as < dlm > characters) is not found in those lines.

In token mode, each line is considered a sequence of tokens. The type of file on which the FIND is being performed is recognized and tokens are then determined based on that file type. A token is defined as follows:

- For ALGOL, DCALGOL, DMALGOL, NDLII, NEWP, and Pascal files, one of the following:
  - Any group of adjacent alphanumeric characters (uppercase and lowercase letters, digits 0 through 9) and underscores ( \_ )
  - A single nonalphanumeric graphic character that is not an underscore or a blank
- For COBOL, COBOL74, and DASDL files, one of the following:
  - Any group of adjacent alphanumeric characters (uppercase and lowercase letters, digits 0 through 9) and hyphens (-)
  - A single nonalphanumeric graphic character that is not a hyphen or a blank
- For all other file types, one of the following:
  - Any group of adjacent alphanumeric characters (uppercase and lowercase letters, digits 0 through 9)
  - A single nonalphanumeric graphic character that is not a blank

Any number of blanks can separate tokens; at least one blank must separate adjacent alphanumeric tokens. For example, the following line illustrates the use of tokens:

```
100 abc, def5-3xyz i j k& *+0 end_token
```

The preceding line contains the following tokens:

ALGOL, DCALGOL, DMALGOL, NDII, NEWP, Pascal Files	COBOL, COBOL74, DASDL Files	Other Files
abc	abc	abc
,	,	,
def5	def5-3xyz	def5
-		-
3xyz		3xyz
i	i	i
j	j	j
k	k	k
&	&	&
*	*	*
+	+	+
@	@	@
end_token	end	end
	-	-
	token	token

The search is successful whenever the same sequence of tokens in the target text is found in a line of the file. More than one token may be included in one <text> construct. As examples, depending on the file type, the following targets might or might not be found in line number 100 in the previous example (periods are used as <dlm> characters):

Target	File Type
.abc.	Found for all file types
., def5-3xyz.	Found for all file types except COBOL and COBOL74
.end_token.	Found for ALGOL, DCALGOL, DMALGOL, NDII, NEWP, and Pascal files only

The following targets would not be found in line number 100 in any file type:

- .a.
- .5.
- .k&.

## FIND Command (cont.)

---

### Examples

```
LIST FILE1
#FILE (UZER)FILE1 ON USERFILES
100 THIS IS A LIST.
200 THIS IS A PAGE.
300 HERE IS SOME CODE.
400 SECOND LIST.
500 END.
#
```

```
FIND /LIST/:T
#WORKFILE FILE1
100 THIS IS A LIST.
400 SECOND LIST.
#
```

```
FIND /LIST/,/PAGE/,/CODE/ 300-END
#WORKFILE FILE1
300, 400
#
```

```
FIND LITERAL /HIS/, LIT /IST/:T
#WORKFILE FILE1
100 THIS IS A LIST
200 THIS IS A PAGE.
400 SECOND LIST.
#
```

```
FIND LIT OCCUR 4 /IS/:T
#WORKFILE FILE1
200 THIS IS A PAGE.
#
```

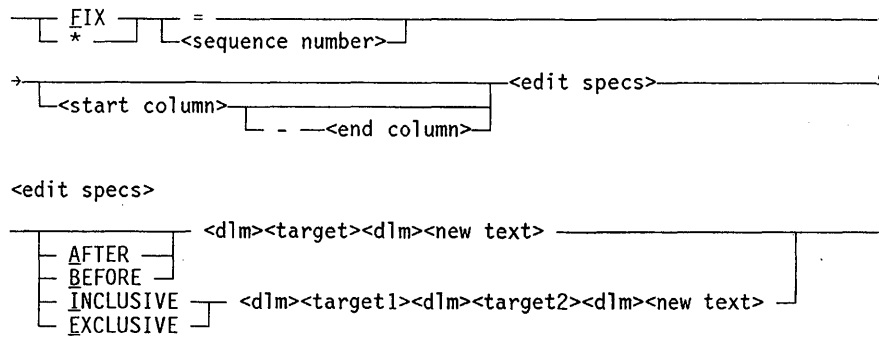
The following example shows the FIND command used with the TEXT option, when some lines were marked with a current MARKID:

```
FIND /LIST/ FILE1:TEXT
#FILE (ESR)FILE1 ON P00G
100*THIS IS A LIST.
400*SECOND LIST.
```



# FIX

## Syntax



## Explanation

The FIX command is used to edit a line of text by inserting new text within the original text, replacing original text with new text, or deleting text.

The basic action of a FIX command is to locate an objective, which can be a character or group of adjacent characters to be deleted or replaced; the objective may also be a point between two characters where an insertion is to be made. When the objective is found, the deletion or insertion is performed. Text to the right of the objective is moved right or left if the insertion is longer or shorter than the deletion; trailing blanks are discarded or inserted as needed. If nonblank characters are pushed past the end of the text field, the FIX operation is performed, but an overflow error is noted.

The asterisk (\*) is a synonym for FIX and may be used to edit lines while in single-line sequencing mode. If the asterisk is the first character after the system-supplied sequence number, CANDE scans the line to determine if it is a valid FIX command. If the line is valid, the command is executed; otherwise, the line is entered as a line of text. The asterisk form of the FIX command cannot be used in page-mode sequencing.

### <sequence number>

The sequence number of the line to be fixed is specified as an integer or by an equal sign (=); an equal sign refers to the last line entered or fixed and is undefined if no entry or FIX has been entered since the work file was updated.

### <start column>

### <end column>

The target search may be bounded or supplanted by column specifications; <start column> and <end column> represent integers; <end column> cannot be less than <start column>, and both must lie within the text field for the work file. Column specifications modify the <edit specs> construct as described at the end of the discussion of the <edit specs> construct.

Note that the FIX command differs markedly from the REPLACE command. The FIX command acts on only one line, but the REPLACE command can act on a range of lines. If a line overflow does occur, FIX performs the action but REPLACE does not. (Both situations generate error messages.)

### <edit specs>

The < new text > construct can contain any sequence of characters and is null if the end-of-line (EOL) immediately follows the last < dlm >. (In a SCHEDULE session, the end-of-line is assumed to follow the last nonblank character on the line, unless an EOL character has been defined and used.)

When a non-null target is specified, CANDE scans the text field of the specified line in the work file until a sequence of characters is found that exactly matches the target. All characters including blanks are significant. This target is to be replaced by the new text (or deleted if the new text is null) unless BEFORE or AFTER is specified. If BEFORE or AFTER appears, the new text is inserted just preceding or just following the target, respectively. If the new text is null when BEFORE or AFTER is specified, no change will be made in the line. If the new text is longer than the target text, an overflow of text beyond the right margin may occur, resulting in an error message. If the target is null, the new text is inserted at the beginning of the line of text.

If < target1 > and < target2 > are specified, and both are non-null, CANDE scans the line of text for a pattern exactly matching < target1 >, and then seeks < target2 > in the remainder of the text field beyond the < target1 >. The new text then replaces the aggregate of the two targets and the text between them (INCLUSIVE) or just the text between them (EXCLUSIVE). If < new text > is null, the line of text does not change. If the new text is shorter than the length of text it is to replace, blanks will replace the remainder of the text bounded by the two targets. If < target1 > is null, it is treated by CANDE as the first character in the line of text, and is always included in the text to be replaced, even if EXCLUSIVE is specified. If < target2 > is null, it is treated by CANDE as the last character in the line of text, and is always included in the text to be replaced, even if EXCLUSIVE is specified.

If column specifications are included in the FIX command, they modify the < edit specs > construct. If < start column > or < start column > - < end column > is specified, then < target >, < target1 > and < target2 > are only sought within the inclusive boundaries of < start column > and the end of line, or < start column > - < end column >, respectively. An error message is issued if the specified target or targets are not found within those boundaries. If < target > or < target1 > is null, then < start column > takes over their function. If < target2 > is null, < end column > functions as < target2 >. Column specifications affect determination of the objective but do not limit the text that may be shifted to effect the fix.

If a hexadecimal escape character is set through the CANDE command ESCAPE, then < target >, < target1 >, < target2 >, and < new text > can contain hexadecimal values delimited by the defined hexadecimal escape character. However, the target delimiter (< dlm >) cannot be the same as the hexadecimal escape character. For more information about the hexadecimal escape character, refer to the ESCAPE command earlier in this section.

The FIX commands are tanked and are executed only when the next UPDATE, LIST, or page-invoking command occurs. Because of tanking, any error messages that result are not displayed until the time of the next UPDATE, LIST, or page-invoking command; therefore, the error messages may not be displayed at the time the command is entered.

The following listing summarizes the many variations of the FIX command. In the following listing, *m* and *n* represent integer column numbers; *x* and *y* represent text strings. A blank space in the listing denotes an absent <start column>, <end column>, or keyword; a hyphen denotes a null target. Where several variations achieve the same effect, only the simplest variation is shown (degenerate cases like BEFORE or AFTER an empty target are omitted). The following listing assumes that the new text is not null. If the new text is null, the word *replace* should be *delete*, and the word *insert* should be *do nothing*.



**Notes:**

- *The target search in a FIX command is satisfied by the first match encountered. Enough context (or a column restriction) must be provided to unambiguously specify the intended objective.*
- *The <dlm> character may not appear in a target, but it may appear in the new text. Inadvertent use of the <dlm> character within the <target> construct is not detected as an error; CANDE processes a shorter <target> and a longer <new text> than is intended.*
- *The FIX command makes a distinction between uppercase and lowercase characters. Use the appropriate uppercase and lowercase characters in the <target>, <target1>, <target2>, and <new text> constructs to perform the intended search and insert, replace, or delete operations.*

**Examples**

```
L
100 ABCDEFGHIJKLMNOPQRSTUVWXYZ
300 IF CARD.PRESENT THEN
400 VIVID HUES OF RED, ORANGE, GREEN AND BLUE
500 X = SQRT (3.1459 * R**4)
#
```

```
FIX 100:FG:HIJ
#
```

```
L =
100 ABCDEHIJHIJKLMNOPQRSTUVWXYZ
#
```

```
FIX = EXCLUSIVE.HIJ..
#
```

```
L 100
100 ABCDEHIJ
#
```

The following example shows a correct change:

```
F 300/CARD.PRESENT/BOOLEAN(CARD.AVAILABLE)
#
```

```
L 300
300 IF BOOLEAN(CARD.AVAILABLE) THEN
#
```



## FIX Command (cont.)

---

The following example shows an incorrect specification for the same change (the period is part of the target text and should not have been used as the delimiter character):

```
F 300.CARD.PRESENT.BOOLEAN(CARD.AVAILABLE)
#

L 300
300 IF PRESENT.BOOLEAN(CARD.AVAILABLE).PRESENT THEN
#

F 400 A.NGE., YELLOW
#

L 400
400 VIVID HUES OF RED, ORANGE, YELLOW, GREEN AND BLUE
#

FIX 400 20-50//
#

L 400
400 VIVID HUES OF RED,
#

FIX 500 E/(/)/AREA
#

L 500
500 X = SQRT (AREA)
#
```

The following example shows the use of the asterisk form from within a single-line sequence mode (note that the target text is not uniquely specified, so the first match is used):

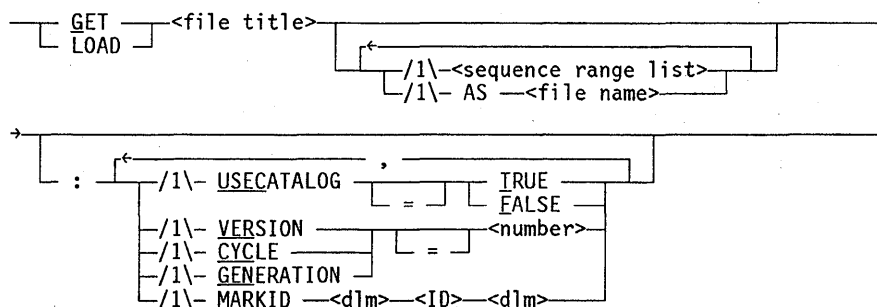
```
M TEST
#WORKFILE TEST:SEQ

S
100THIS IS A TEST
200* 100/IS/ISNT
300
#

L
100 THISNT IS A TEST
#
```

## GET or LOAD

### Syntax



### Explanation

The GET command designates an existing saved file, or portions thereof, as the source for a new work file. LOAD is a synonym for GET. If an unsaved work file exists, the command is ignored and the following message is displayed:

```
#REMOVE OR SAVE WORKFILE
```

The work file must be removed or saved. To specify a new work file, reenter the GET or LOAD command.

The < file title > construct can specify any file to which the user is permitted access. If desired, a < sequence range list > can be specified, causing only a portion or portions of the file to be used as the source.

The name of the work file to be created can be specified by using the AS < file name > option; otherwise, the work file name is a temporary copy of the file title.

The GET command might be unable to retrieve a file as a work file if the file name is not in the user's directory and the AS < file name > option is absent. A file is not in the user's directory when the file name is preceded with an asterisk (\*) or another's usercode, or an ON clause specifies a family name for the file. When you use the GET command to retrieve a file that is not in your directory, that file is considered a *foreign file*. That is, the retrieved file is not a work file. The following message is an example of what is displayed when a file is retrieved as a foreign file:

```
#WORKFILE IS NOT NAMED; SOURCE IS *LEDGER/JANUARY ON GL: SEQ,
27221 RECORDS
```

When the LOCKEDFILE attribute is set to TRUE for a file, that file is retrieved as a foreign file because that file cannot be removed and its file name cannot be changed. For more information about the LOCKEDFILE file attribute, refer to the *File Attributes Reference Manual*. See the ALTER command earlier in this section for information about changing the LOCKEDFILE attribute.



For installations that use file cataloging, the GET command allows specification of USECATALOG, VERSION, CYCLE, and GENERATION. For an explanation of these attributes, refer to the *File Attributes Reference Manual*.

The *MARKID* <dlm> <ID> <dlm> option establishes an initial MARKID. The MARKID command can still be used at any time afterwards to establish a new MARKID.

The following commands display the correspondence between the ID field of a record and the current value of the MARKID, if the MARKID option is specified:

- FIND on a work file using the TEXT option
- RANGE on a work file using the TEXT option
- LIST on a work file without the UNSEQUENCED or PUNCH options
- REPLACE using the TEXT option, before the record ID field is replaced

The displays from each of the preceding commands include the following:

- The sequence number
- An asterisk (\*) if the record ID is the same as the MARKID; otherwise, a blank
- The text

### Examples

```
GET AFILE
#WORKFILE AFILE: SEQ, 5 RECORDS, SAVED
```

```
LOAD TERMPAPER 400-END AS INDEX
#WORKFILE INDEX; SOURCE IS (UZER)TERMPAPER ON USERPACK: DATA
```

```
GET (ANYUSER)ANYFILE ON HISPAC AS MYWORKFILE 0-150,400-END
#WORKFILE MYWORKFILE; SOURCE IS (ANYUSER)ANYFILE ON HISPAC: ALGOL
```

```
GET EXAMPLE: MARKID 10/01/81_
#WORKFILE EXAMPLE: ALGOL,500 RECORDS, SAVED, MARKID "10/01/81 "
```

## GET Command (cont.)

---

The following examples show the results of attempts to retrieve files where the LOCKEDFILE file attribute has either been set or reset.

- The following example retrieves the file LEDGER/JUNE that is in the user's directory and the LOCKEDFILE file attribute is reset to FALSE.

```
GET LEDGER/JUNE
#WORKFILE LEDGER/JUNE: SEQ, 23000 RECORDS, SAVED
```

- The following example retrieves the file LEDGER/JUNE that is in the user's directory (ACCT) as a workfile named LEDGER/2NDQTR. The LOCKEDFILE file attribute is reset to FALSE.

```
GET LEDGER/JUNE AS LEDGER/2NDQTR
#WORKFILE LEDGER/2NDQTR; SOURCE IS (ACCT)LEDGER/JUNE ON PACK: SEQ,
23000 RECORDS
```

- The following example shows an attempt to retrieve the file LEDGER/JUNE that is in the user's directory (ACCT) as a workfile. The LOCKEDFILE file attribute is set to TRUE.

```
GET LEDGER/JUNE
#WORKFILE IS NOT NAMED; SOURCE IS (ACCT)LEDGER/JUNE ON PACK(LOCKEDFILE):
SEQ, 23000 RECORDS
```

- The following example retrieves the file LEDGER/JUNE that is in the user's directory (ACCT) as a workfile named LEDGER/COPY. The LOCKEDFILE file attribute is set to TRUE.

```
GET LEDGER/JUNE AS LEDGER/COPY
#WORKFILE LEDGER/COPY; SOURCE IS (ACCT)LEDGER/JUNE ON PACK(LOCKEDFILE):
SEQ, 23000 RECORDS
```

# HELLO

## Syntax

```

HELLO <usercode>
      .
      /
      <password>
  
```

## Explanation

The HELLO command terminates the current session and initiates a new session after log on is complete. Dial-up lines are not disconnected.

When HELLO is entered, the current session ends and CANDE displays system resource usage statistics, followed by the usual greeting, which contains

- System name and serial number
- CANDE MCS name (usually CANDE) and version level
- System hostname
- Station name and LSN

System resource usage statistics are provided in the form of the session number, elapsed time of session (ET), processor time consumed (PT), I/O time consumed (IO), usercode, current time, and the date. They are printed in the following format:

```

#END SESSION <session number> ET=<time> PT=<time> IO=<time>
#USER = <usercode> <time> <date>
  
```

A chargecode and accesscode are also printed if applicable. A prompt is then issued to enter a usercode.

When *HELLO* *<usercode>* *<password>* is entered, the log-on procedure is completed (an accesscode and chargecode might also be requested to log on). If the *<password>* is not entered, the system requests this information in the normal log-on manner.

On InfoGuard systems with the SECOPT CLASS set to S1 or higher, the password must be entered on a separate line. Refer to the *Security Administration Guide* for additional information about security options.

If an unsaved work file exists when the HELLO command is entered, CANDE displays the following message:

```
#REMOVE OR SAVE WORKFILE
```

The user must save or remove the work file and re-enter the HELLO command before CANDE will terminate the session.

The *<usercode>* . form is used to log on to a usercode that does not have a password.



For additional information about usercode/passwords and logging on, refer to "User Identification and Logging On" in Section 1, "General Information."

**Examples**

```
HELLO UZER/PSW
#END SESSION 0383 ET=2:13:35.7 PT=21.7 IO=43.4
#USER = PAYROLL CHARGE = 4736. 10:05:12 10/20/84
```

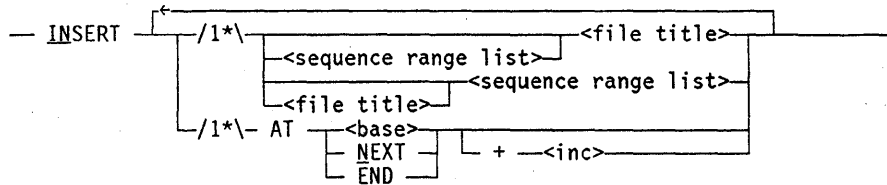
```
#B7900:7 CANDE 36.120 AT SITE7; YOU ARE TD12345(83)
#SESSION = 0824 USER = PAYROLL CHARGE = 4736
```

```
hello userid
#ENTER PASSWORD PLEASE
```

```
HELLO USERID PASSID
#SESSION 3975 16:30:56 04/20/78
```

## INSERT

### Syntax



### Explanation

The INSERT command copies lines from a file (by default, the work file) and places the copies in the work file as a contiguous block with new sequence numbers.

An UPDATE is implicitly invoked after the INSERT has been executed.

The < file title > construct names the file to be inserted and can be any file to which the user is allowed access. The work file is assumed if no < file title > is specified.

The < sequence range list > construct specifies the sequence range or ranges of the lines to be copied into the work file or < file title >.

The new sequence numbers of copied lines are determined by assigning an initial value to the first line and incrementing that value for each succeeding line. The initial value must be specified in one of three ways:

- The < base > specifies the value explicitly. The value of < base > must be an < integer >.
- The NEXT option sets the initial value to the next sequence number that would have resulted from the most recent MOVE, INSERT, RESEQ, or SEQ command. A default value of 100 is used if no such command has appeared since the last MAKE, GET, or DELETE ALL command.
- The END option sets the initial value to the largest sequence number currently in the file, plus the current specified (or default) increment.

The < inc > construct defines the increment value to use for successive sequence numbers in the block. The value of < inc > must be an < integer >. If the increment value is unspecified, the increment value from the most recent MOVE, INSERT, RESEQ, or SEQ command is used. (100 is used if none of these commands have appeared.)

The range of new sequence numbers cannot overlap any lines already in the file, nor can the numbers exceed the largest sequence number that may be expressed in the sequence number field.

**Examples**

```
LIST
#WORKFILE TESTONLY
100 LINE 1
200 LINE 2
300 LINE 3
400 LINE 4
#
```

```
INSERT 100-300 AT END+10
#UPDATING
#
```

```
LIST
#WORKFILE TESTONLY
100 LINE 1
200 LINE 2
300 LINE 3
400 LINE 4
410 LINE 1
420 LINE 2
430 LINE 3
#
```

```
INS VEGETABLES AT 402+2
#UPDATING
#
```

```
INSERT AT END FRUIT
#UPDATING
#
```

```
LIST
100 LINE 1
200 LINE 2
300 LINE 3
400 LINE 4
402 CARROT
404 TOMATO
406 EGGPLANT
410 LINE 1
420 LINE 2
430 LINE 3
432 BANANAS
434 ORANGES
436 APPLES
```

# LANGUAGE

### Syntax

— LANGUAGE \* <language name>

<language name>

—<letter> /16\ <alphanumeric character>

### Explanation

The LANGUAGE command alters the language specification of the current session.

When LANGUAGE is entered, CANDE displays the language currently in effect for the session.

The *LANGUAGE \** form restores the current language for the session to the default language. The default language specification is obtained by CANDE from the USERDATAFILE. If no language is specified for the user in the USERDATAFILE, the system language is used.

The *LANGUAGE <language name>* form changes the language for the session to the language represented by the <language name>.

Refer to the *Security Administration Guide* for information on changing the default language specifications in the USERDATAFILE.

### Examples

```
LANGUAGE  
#LANGUAGE: ENGLISH
```

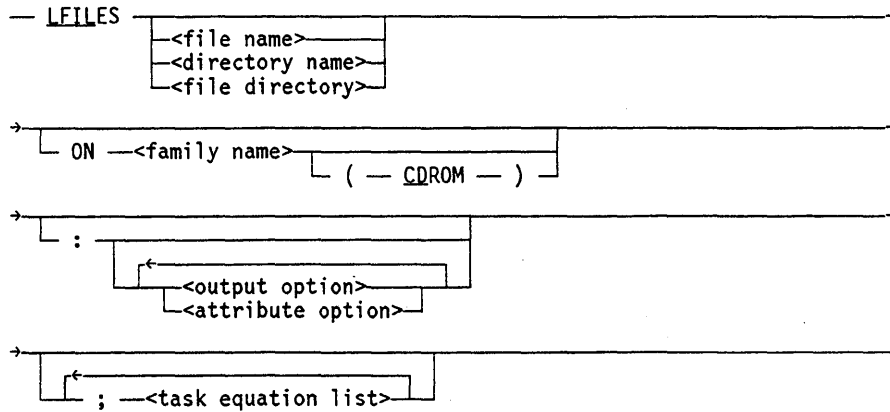
```
LANGUAGE FRANCAIS  
#LANGUAGE CHANGED FROM ENGLISH TO FRANCAIS
```

```
LANGUAGE *  
#LANGUAGE CHANGED FROM FRANCAIS TO ENGLISH
```

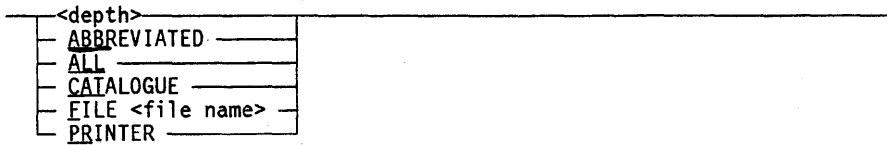


# LFILES

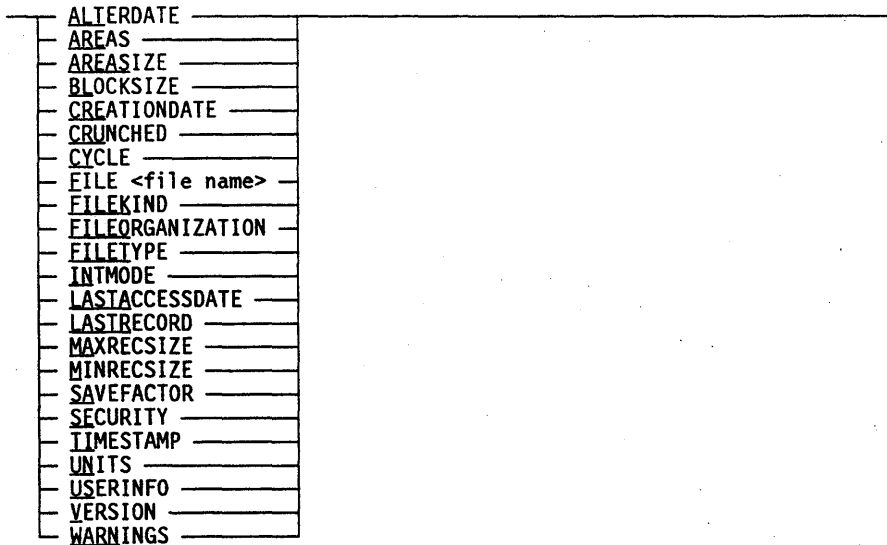
## Syntax



### <output option>



### <attribute option>



## Explanation

The LFILES command executes the utility SYSTEM/FILEDATA, which lists the names and file attributes of files. If a <file name> or <directory name> is not specified, the names and attributes of all files in the user's library are listed.

The <file name> specifies a file for which file attribute information is desired. The <file name> can refer to any file to which the user is allowed access. Note that if ON <family name> is specified, family substitution does not apply. If ON <family name> is followed by CDROM, the family name refers to a CD-ROM reader rather than a pack or disk. A CD-ROM reader is a read-only device that reads files stored on a compact disc.

The <directory name> specifies a directory and results in a list of the file attribute information for each file subordinate to that directory. However, information is printed only for those files to which the user is allowed access.

If both a file and a directory exist for the name supplied, information is listed for both the file and the directory (plus the subordinate files of the directory).

The amount of file attribute information to be displayed depends on whether available options are set or reset. By default, all information regarding file attributes is displayed (that is, all options are set). The amount of file attribute information to be displayed can be restricted by naming attribute options and using the available output options. Options are recognized after the appearance of the colon (:). The colon resets all file attribute options except FILEKIND; the file attribute options can then be set selectively.

Some of the file attribute option names are identical to the corresponding file attribute names but print more information than the file attributes contain.

The file attribute options ALTERDATE, CREATIONDATE, and LASTACCESSDATE print both the date and the time.

When the WARNINGS option is specified, the warnings that have been accumulated by a file are expanded into their corresponding text and reported.

For further information about file attributes, refer to the *File Attributes Reference Manual*.

The available output options are discussed in the following paragraphs.

The <depth> construct must be an <integer>. This option restricts the depth of listing for multilevel file names.

The ABBREVIATED option causes the specified file attribute names to be abbreviated in the output.

The ALL option causes information for all file attributes, except USERINFO, to be listed. This option is assumed if no options are specified.

The CATALOGUE option reports the existence of nonresident, catalogued files.

The FILE option causes the output to be written to a file with the specified name.

The PRINTER option causes the output to be sent to the line printer.

**Examples**

```
LFILES BREAKER ON USERFILES
#RUNNING 2497
#?
(TERRI) : DIRECTORY ON USERFILES
. BREAKER : ALGOLSYMBOL AREAS=15 AREASIZE=450 BLOCKSIZE=450
          CREATIONDATE= 2/12/81 @ 13:19:15 CYCLE=1 FILETYPE=0
          INTMODE=4 LASTACCESSDATE= 2/12/81 @ 13:19:15
          LASTRECORD=12 (225 SEGS) MAXRECSIZE=15 MINRECSIZE=0
          SAVEFACTOR=30 SECURITY=PRIVATE (I/O)
          TIMESTAMP= 2/12/81 @ 13:19:17 UNITS=0 VERSION=0
          ALTERDATE= 2/12/81 @ 13:19:15
#
```

```
LFILES (JOHNSON)ACCOUNT/SUMMARY:BLOCKSIZE AREASIZE MAXRECSIZE ABB
#RUNNING 2513
#?
(JOHNSON) : DIRECTORY ON USERPACK
. ACCOUNT : DIRECTORY
. . SUMMARY : SEQDATA ASIZE=420 BLK=420 MAX=14
#
```

The following example of the LFILES command lists the file attributes of the file README.DOC on a CD-ROM:

```
LFILES "README.DOC" ON PROGLIB_1A(CD)
#RUNNING 5693
#?
*"README.DOC" : DATA ALTERDATE=06/30/1989 @ 01:01:00
                AREALENGTH=15220 (15220 RECORDS) AREAS=1 AREASECTORS=8
                BLOCKSTRUCTURE=FIXED CREATIONDATE=06/30/1989 @ 01:01:00
                CYCLE=1 EXTMODE=OCTETSTRING FILELENGTH=15220
                FILEORGANIZATION=NOTRESTRICTED FILESTRUCTURE=STREAM
                FRAMESIZE=8 LASTACCESSDATE=06/30/1989 @ 01:01:00
                LASTRECORD=15219 MAXRECSIZE=1 MINRECSIZE=1 SAVEFACTOR=1
                SECURITY=PUBLIC (IN) TIMESTAMP=06/30/1989 @ 01:01:00
                TOTALSECTORS=8 VERSION=0
```

# LIST

## Syntax

— LIST — <specified file syntax> — <work file syntax> —

### <specified file syntax>

— /1\*\-<file title> —  
 — /1\-<sequence range list> —  
 — /1\-<sequence number> FOR <integer> —  
 — /1\- @ <start column> - - <end column> —  
 — /1\-<altered records options> —

### <altered records options>

— : — A —  
 — AF —  
 — AFN —  
 — AN —  
 — ANF —  
 — CHANGES —  
 — COMPARE —  
 — FLAG —  
 — PAGEFORMAT —  
 — PUNCH —  
 — SQUASHED —  
 — TRUNCATED —  
 — UNSEQUENCED —

### <work file syntax>

— /1\-<sequence range list> —  
 — /1\-<sequence number> FOR <integer> —  
 — /1\-<current line handling> —  
 — /1\- @ <start column> - - <end column> —  
 — /1\-<altered records options> —

### <current line handling>

— = — > <integer> — FOR — <integer> —  
 — < <sequence number> — = —

## Explanation

The LIST command displays the contents of a work file or other specified file at the station. If a file title is not specified, the contents of the work file are listed. If a file title is specified, the file is listed provided that the user has access privileges.

### Listing a Specified File

The LIST command displays a file other than the work file if a file title is specified. If the user has sufficient file access privileges, the specified file need not be limited to the user's files.

LIST <sequence range list>

A sequence range list in the command indicates that only the records in the specified sequence range are to be listed.

LIST <sequence number> FOR <integer>

The *LIST* <sequence number> *FOR* <integer> form of the command lists a specified number of lines starting from the sequence number. The *FOR* <integer> construct specifies the total number of lines to be listed. If the number of lines from the sequence number to the end of the file is less than the specified integer, the LIST option displays the number of lines available in the file. If the integer specified is 0 (zero) or 1, then the sequence number is listed. If the sequence number does not exist in the file, the list starts with the next higher sequence number, if any.

LIST @ <start column>

LIST @ <start column> - <end column>

The @ <start column> option lists records beginning from the specified start column. The *LIST* @ <start column> - <end column> form of the command can be used to indicate that only the specified range of columns of each record are to be listed. The start column and end column values must be integers, the end column value cannot be less than the start column value, and both the start column and the end column values must be within the text field of the file.

### Altered Records Options

The following LIST options are used to list alterations that have been made to the work file by FIX, DELETE, and single-line entry since the work file was last updated. (An update is performed when an UPDATE command is entered or when an update is forced by the MERGE, RMERGE, EXCLUDE, INSERT, MOVE, RESEQ, COMPILE, EXECUTE, TYPE, REPLACE, UTILITY, or SAVE command. The LIST command does not cause an update.)

#### A

The A option lists only altered lines in the work file. The original version of each line that has been deleted by a DELETE command or changed by a single-line entry is listed and flagged with a minus sign (-). The original versions of lines modified by the FIX command are not listed. New lines of text are flagged as follows:

- For an old line replaced by a new line, the flag is an *R*.
- For a new line inserted, the flag is an *I*.

- Each line that has been modified by the FIX command is listed and flagged with an *F*.

Blank lines are inserted to separate the listing into groups; a group contains the original version of a line of text (except for lines changed by the FIX command) followed by the changed, replaced, or inserted line of text. Consecutively inserted or deleted lines are shown as a single group.

### AF

The output from the AF option is the same as the A option output, except that the original version of lines changed by the FIX command are also displayed. The line showing the record before the FIX command was transmitted is flagged with a minus sign (-). (This option is functionally equivalent to the COMPARE option.)

### AN

The output for the AN option is the same as the A option output, except that the neighboring lines of a new line are listed and are not flagged.

### AFN, ANF

The AFN option combines the effects of the AN and AF options. Lines are shown before fixing, and the neighbors of new lines are shown. AFN and ANF are synonymous.

### CHANGES

The output for the CHANGES option is the same as the A option output, except that only the sequence numbers (not the text) of deleted lines appear in the list, and groups of altered lines are not separated by vertical lines.

### COMPARE

The COMPARE option lists only altered lines in the work file. The original version of each line that has been deleted by the DELETE command, or changed by a single-line entry or a FIX command is listed and flagged with a minus sign (-). New lines of text are flagged as follows:

- For an old line replaced by a new line, the flag is an *R*.
- For a new line inserted, the flag is an *I*.
- Each line that has been modified by the FIX command is listed and flagged with an *F*.

Blank lines are inserted to separate the listing into groups; a group contains the original version of a line of text followed by the fixed, replaced, or inserted line of text. Consecutively inserted or deleted lines are shown as a single group. The COMPARE option is functionally equivalent to the AF option.

**FLAG**

The FLAG option causes each line in the work file to be listed with the altered lines flagged as described for the A option. Groups of altered lines are not separated by vertical spaces.

**Format Options**

Several output format options are available with the LIST command. If no format option is specified, the default format is used; that is, the sequence number of the line appears (pseudo-sequence numbers for type data files) with leading zeros omitted, followed by a blank and the text of the line. Lines too long for the terminal are split (on token boundaries where possible). All lines that have been marked with a current MARKID are displayed with an asterisk (\*) between the sequence number and the rest of the line, as long as the PUNCH and UNSEQUENCED options are not in effect. For more information on the MARKID, refer to the GET, MAKE, and MARKID commands. The format options and their effects are described in the following paragraphs.

**PAGEFORMAT**

The PAGEFORMAT option causes the output to appear in a format similar to the page mode output. That is, the full length of the sequence number field is displayed, immediately followed by the contents of the text field (the blank or asterisk flag character is omitted).

**PUNCH**

The function of the PUNCH option is to punch a paper tape on the paper tape punch of a terminal so equipped. The system waits five seconds for the user to turn on the paper tape punch if the punch is not in AUTO-START. The system sends a DC1 (XON) character followed by 10 NULs, and the name of the file (or the work file name if a file title is not specified), followed by 40 additional NULs. Each line of data ends with a carriage return, a line feed, DC1 character, and a NUL. A DC3 (XOFF) character is punched at the end of the tape. The tape can be read back to the system using the TAPE command by positioning the tape in the reader in the area of the 40 NULs. Sequence numbers punched include leading zeros.

**SQUASHED**

The SQUASHED option causes any group of multiple blanks to be reduced to a single blank.

**TRUNCATED**

The TRUNCATED option causes the output line to be truncated if the width of the CANDE station is less than the MAXRECSIZE of the work file. (The continuation character designates the character position at which truncation takes place.)

### UNSEQUENCED

The UNSEQUENCED option causes the lines to be listed without sequence numbers.

### Listing a Work File

If a file title is not specified in the LIST command, the user's work file is displayed as specified by the LIST options. The LIST options for a work file are identical to those for a specified file, except that they include the added feature of listing lines of the work file from the *current* line.

### <current line handling>

The current line of the work file is defined as one of the following:

- The line that was last entered.
- The line that was last modified with the FIX command.
- The line that was specified through the LIST command that sets the current line. For example, line 800 is the current line if the following form of the LIST command is used:

```
LIST 800 =
```

The last action entered determines the current line. The concept of the current line and the LIST options for manipulating the current line apply only to editing work files.

LIST =

The *LIST =* form of the command lists the current line of the work file.

LIST > <integer>

LIST > <integer> FOR <integer>

The *LIST > <integer>* form of the command moves the file forward the number of lines specified from the current line and displays that line number. The new line becomes the current line. The *FOR <integer>* option displays the number of lines specified after the new current line.

LIST < <integer>

LIST < <integer> FOR <integer>

The *LIST < <integer>* form of the command moves the file backwards the number of lines specified from the current line and displays that line number. The new line becomes the current line. The *FOR <integer>* option displays the number of lines specified after the new current line.



**Examples**

```
1
100 BEGIN
200 REAL Y;
300 Y:=2;
400 END.
#
```

```
LIST (UZER)FILE ON USERPACK 200-400 @72-80
#FILE (UZER)FILE ON USERPACK
200 THIS IS
300 COMMENT
400 SECTION
#
```

The following example lists ten lines starting at line 1500:

```
LIST 1500 FOR 10
```

The following example lists 21 lines starting at line 200100:

```
L 200100 F 21
```

The following example lists four lines that have the current MARKID:

```
L 40-70
40*Now is the time
50*for all good people
60*to come to the aid
70*of their country.
#
```

## LIST Command (cont.)

---

The following example lists the full sequence number field followed by the contents of the text fields:

```
LIST 1,999000:PA
00000001This is my very first line
00999000... and this is the last
```

The following examples demonstrate the use of the LIST command when working with a work file:

```
L
100 Line 1
200 Line 2
300 Line 3
400 Line 4
500 Line 5
600 Line 6
700 Line 7
#

L 400 =
400 Line 4

FIX = .4.4mod
#

L =
400 Line 4mod
#

L > 2 FOR 2
600 Line 6
700 Line 7
#

L =
600 Line 6
#

L < 3 FOR 3
300 Line 3
400 Line 4mod
500 Line 5
#
```

## **LOAD**

The **LOAD** and **GET** commands are synonyms. (Refer to the **GET** command in this section.)

**LOAD Command** (cont.)

---

# LOG

## Syntax

```
-- LOG <option list> ; --<task equation list>
```

## Explanation

The LOG command executes the LOGANALYZER utility. This utility extracts specific information, as specified by the <option list> construct, from the system log file and displays the information at the user's terminal. Output from this command is, by default, directed to the remote terminal.

The <option list> construct must be a string of standard LOGANALYZER options. The <option list> may be omitted, indicating the ALL option; however, the ALL option produces voluminous output.

LOGANALYZER is documented in the *A Series System Software Support Reference Manual*. The syntax and explanation for <option list> and examples of output from the LOG command are given in that manual.

## Examples

```
LOG MIX 6483
```

```
LOG PRINTER ALL
```

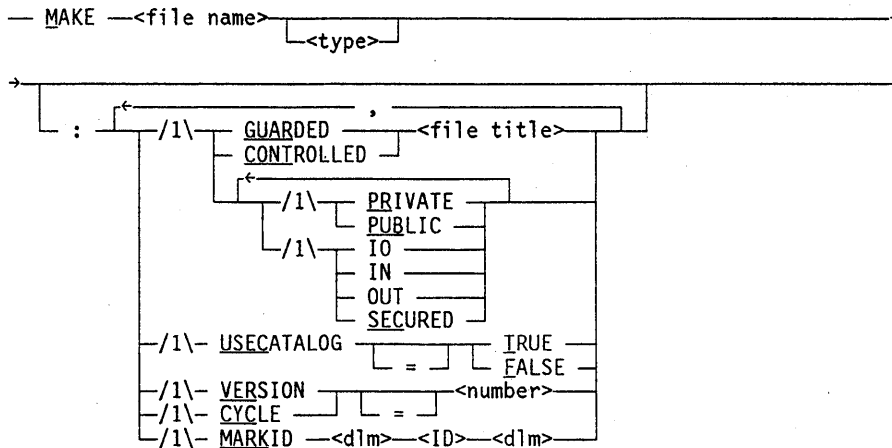
```
LOG ALL CREATE <file name>
```

```
LOG ALL APPEND <file name>
```

```
LOG ALL;FILE LINE(KIND=PRINTER)
```

# MAKE

## Syntax



## Explanation

The MAKE command creates a new work file. If an unsaved work file already exists, the following message is displayed:

```
#REMOVE OR SAVE WORKFILE
```

The work file must be removed or saved before another work file can be created.

The < file name > defines a new file within the user's library to become the name associated with the work file. If the file < type > is not specified, the work file is type SEQ by default. Allowable types and their associated attributes are defined in the "Basic Constructs" section.

The < file name > cannot reference an existing file in the user's library. CANDE rejects such requests with a message similar to the following:

```
#SOURCE AND OBJECT FILE ALREADY PRESENT
```

The security of the work file can be specified with the PUBLIC, PRIVATE, GUARDED, CONTROLLED, SECURED, IN, OUT, or IO options. The default security is PRIVATE IO. (CLASSA and CLASSB are recognized as nonpreferred synonyms for PUBLIC and GUARDED, respectively.) For further information on the security specifications listed above, refer to the SECURITYGUARD and SECURITYTYPE file attributes in the *File Attributes Reference Manual*.

If a guard file is specified, its name must be less than 137 characters, including a prefix of either a usercode or an asterisk (\*) prefix, and an ON < pack name > suffix. The prefix and suffix are supplied by CANDE if not specified by the user.

The USECATALOG option can be specified for those installations that use file cataloging. The USECATALOG option can be used only when the CANDE option

CATALOGOK is set (see the *A Series CANDE Configuration Reference Manual* for additional information about CATALOGOK).

The *MARKID* <dlm> <ID> <dlm> option establishes an initial MARKID. The MARKID command can be used at any time afterwards to establish a new MARKID.

The following commands display the correspondence between the ID field of a record and the current value of the MARKID, if the MARKID option is specified:

- FIND on a work file using the TEXT option
- RANGE on a work file using the TEXT option
- LIST on a work file without the UNSEQUENCED or PUNCH options
- REPLACE using the TEXT option, before the record ID field is replaced

The displays from each of the preceding commands include the following:

- The sequence number
- An asterisk (\*) if the record ID is the same as the MARKID; otherwise, a blank
- The text.

### Examples

```
make it
#WORKFILE IT: SEQ
```

```
MAKE TOP/SECRET : PRIVATE,OUT
#WORKFILE TOP/SECRET: SEQ
```

```
MAKE CAT/FILE: USEC=TRUE,VERSION=5,CYCLE=11
#WORKFILE CAT/FILE: SEQ
```

```
MAKE GUARDED/PROGRAM COBOL: GUARDED FBI ON TRIAL
#WORKFILE GUARDED/PROGRAM: COBOL
```

```
make testonly a
#WORKFILE TESTONLY: ALGOL
```

## MAKE Command (cont.)

---

```
MAKE CANDE/EX ALGOL:MARKID "ORIG"  
#WORKFILE CANDE/EX : ALGOL, MARKID "ORIG      "
```

```
100XXX  
200YYY
```

```
LIST  
100*XXX  
200*YYY  
#
```

```
SAVE  
#UPDATING  
#WORKSOURCE CANDE/EX SAVED
```

```
GET CANDE/EX : MARKID "CHGED"  
#WORKFILE CANDE/EX: ALGOL, 2 RECORDS, SAVED, MARKID "CHGED  "
```

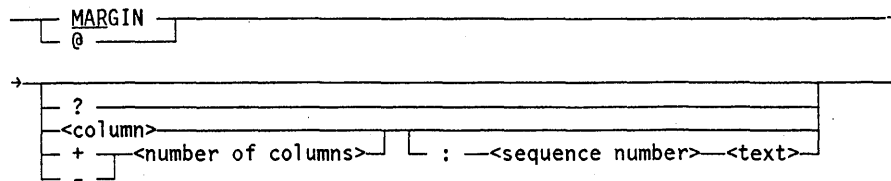
```
100XXXABC
```

```
LIST  
100*XXXABC  
200 YYY  
#
```



# MARGIN

## Syntax



<number of columns>

<integer>

## Explanation

The MARGIN command controls text entry at the left margin of a line by inserting a specified number of blanks to the left of the text being entered. If MARGIN is entered, the margin is set to column 1. If *MARGIN ?* is entered, the current margin setting is displayed.

If the <column> option is specified, <column> represents an absolute column number where the text is to start. If the + <number of columns> or - <number of columns> option is specified, <number of columns> is relative to the previous margin setting.

If a colon (:) is specified, the margin specification applies only to the single-line entry that follows.

Margin specifications can be specified in single-line sequencing mode by using the at sign (@) as the first character after the system-supplied sequence number. The margin specification applies only to the line on which it appears. The : <sequence number> option cannot be used with the at sign (@). The @ form of the MARGIN command cannot be used in page mode.

A line overflow may occur as a result of indentation. A message is displayed, and the resulting line is truncated.

## MARGIN Command (cont.)

---

### Examples

```
MAR 20 : 100 IF A > 99 THEN
```

```
@+4  
#
```

```
MARGIN ?  
#MARGIN @5
```

```
200%TEST COLUMN
```

```
L  
100 IF A > 99 THEN  
200 %TEST COLUMN  
#
```

# MARKID

## Syntax

```
— MARKID <dlm> <ID> <dlm>
```

## Explanation

The MARKID command can be used to place a value in the ID field of new or modified records. This command affects only the current work file. Any line in the work file that is changed receives the current MARKID value in its ID field. (Refer to Table 2-1 in this manual for a description of the ID field.)

When the user changes the value of the MARKID, only subsequent changes in the work file will contain the new MARKID. Records updated prior to the MARKID change retain the old MARKID. A record is not considered changed if only the sequence number is changed with the RESEQUENCE command or if the user only interrogates the setting of the MARKID.

If MARKID is entered, CANDE displays one of the following messages:

- #FILE TYPE CANNOT USE MARKID  
This display indicates that the FILETYPE does not have an ID field.
- #MARKID IS NOT IN USE  
This display indicates that no MARKID has been established for the work file.
- #MARKID <ID>  
This display indicates that the value of the MARKID is <ID>.

The *MARKID* <dlm> <ID> <dlm> command form replaces the old value of the MARKID (if any) with the new value specified by <ID>. If this command form is entered, CANDE displays one of the following messages:

- #FILE TYPE CANNOT USE MARKID  
This display indicates that the FILETYPE does not have an ID field.
- #MARKID CAN ONLY BE <number> CHARACTERS LONG  
The <ID> was too long.
- #MARK <ID>  
This display indicates that the value of the MARKID is <ID>.

## MARKID Command (cont.)

---

The MARKID - command form removes the MARKID from use. If this command form is entered, CANDE displays one of the following messages:

- #MARKID IS NOT NOW IN USE  
This display indicates that a MARKID was in use before the MARKID - command was entered, and has been cancelled.
- #MARKID IS NOT IN USE  
This display indicates that no MARKID was in use before the MARKID - command was entered.

If a MARKID is in use, several commands display the correspondence between the ID field of a record and the current value of MARKID. The following commands are included:

- FIND with FILE and TEXT options
- RANGE with FILE and TEXT options
- LIST of the work file without UNSEQUENCED or PUNCH options
- REPLACE with TEXT option before the record ID field is replaced

The display from these commands includes the following information:

- The sequence number
- An asterisk (\*) if the record has the current MARKID; otherwise, a blank
- The text

**Examples**

```
MAKE EXAMPLE ALGOL:MARK /FIRST/  
#WORKFILE EXAMPLE: ALGOL, MARKID "FIRST"
```

```
100BEGIN  
200 REAL I, K;  
300 FILE REM(KIND=REMOTE);  
400 READ(REM, /, I, K);  
500 WRITE(REM, <"DIVIDE = " U>, I/K);  
600END.
```

```
MARK /SECOND/  
#MARKID "SECOND"
```

```
FIX 400/R/WHILE NOT R  
#
```

```
FIX 400/;/ DO  
#
```

```
FIX 500/W/ W  
#
```

```
REPLACE /K//J/:T  
#WORKFILE EXAMPLE  
#UPDATING  
200 REAL I, J;  
400* WHILE NOT READ(REM, /, I, J) DO  
500* WRITE(REM, <"DIVIDE = " U>, I/J);  
#  
L  
100 BEGIN  
200* REAL I, J;  
300 FILE REM(KIND=REMOTE);  
400* WHILE NOT READ(REM, /, I, J) DO  
500* WRITE(REM, <"DIVIDE = " U>, I/J);  
600 END.  
#
```

## MARKID Command (cont.)

---

MARK-  
#MARKID IS NOT NOW IN USE

L @1-90:U

BEGIN

REAL I, J;

FILE REM(KIND=REMOTE);

WHILE NOT READ(REM, /, I, J) DO

WRITE(REM, <"DIVIDE = " U>, I/J);

END.

#

00000100FIRST

00000200SECOND

00000300FIRST

00000400SECOND

00000500SECOND

00000600FIRST

MARK

#MARKID IS NOT IN USE

GET SYM/MINE:MARK .81 MAY 8.

#WORKFILE SYM/MINE: ALGOL, 756 RECORDS, SAVED, MARKID "81 MAY 8 "

FIX 59450/BEGON/BEGIN

#

61400 REAL

MARKID .81 MAY 11.

#MARKID "81 MAY 11 "

61470 RECCS,

MARK

#MARKID "81 MAY 11 "



## MATCH Command (cont.)

---

contains more records than the old file, all extra records in the new file are flagged as inserted lines.

Any two files that are compared with the MATCH command must have the same file type.

If a <sequence range list> is specified, the files are compared only within that sequence range. If a column specification is used (for example, @ <start column>), the records are compared within that column range. If the SEQ option is used, the indicated column range is used as sequence numbers for the comparison. The width of this sequence number cannot exceed eight digits, and the option can be used only with DATA files.

The results of the comparison are formed according to the output options used. By default, output is directed to the terminal and lists only the sequence numbers. Line deletions are preceded by a minus sign (-), line replacements by an *R*, and line insertions by an *I*.

The FILE <file name> option directs output to the specified file. In that file, line deletions are preceded by a minus sign (-), replacements by an *R*, and insertions by an *I*. The resultant file is of type CDATA by default. However, if the RESULT option is used, the output file is the same file type as that of the files being compared. Files created with the MATCH command are unconditionally crunched. If a file entitled <file name> already exists in the user's library, an error message is displayed.

If the COMPARE option is specified, a file named <file name> is created that contains the nonmatching lines in the two files being compared. The comparison file contains the original version of each line in the old file that has been deleted by a DELETE command or changed by a single-line entry or FIX command in the new file. The comparison file also contains new lines of text in the new file flagged as follows:

- Each old line replaced by a new line is flagged with an *R*.
- Each new line inserted is flagged with an *I*.
- Each line that has been modified by a FIX is listed and flagged with an *F*.

Blank lines are inserted to separate the listing into groups; a group contains the original version of a line of text followed by the new (fixed, replaced or inserted) line of text. Consecutively inserted or deleted lines are shown as a single group.

If the RESULT option is used, the result file is the same file type as that of the files compared with the MATCH command.

When the RESULT option is used for language source files (for example, ALGOL and FORTRAN), line insertions and replacements are put in the result file. Single-line deletions are replaced by a compiler control record (CCR)—that is, an otherwise blank record with a dollar sign (\$) in column one. Multiple-line deletions in sequence cause the output of a \$SET VOIDT record at the starting sequence number and a \$POP VOIDT at the ending sequence number. Most compilers recognize these records as deletions. Refer to the appropriate language reference manual for more information about compiler control records (CCRs) and compiler control options.



For DATA file comparisons using the sequence number column specification, line replacements and insertions are put in the result file. For deletions, a blank record is put in the file with sequence number intact.

When records differ in DATA file comparisons not using the sequence number column specification, the new file record is placed in the result file. If the new file is longer than the old file, trailing unmatched records are placed in the result file. If the old file is longer than the new file, the trailing unmatched records are not placed in the result file.

If the EQUAL option is used, equal records are placed in the output.

The TEXT option directs output to the terminal using the default specifications.

If the SQUASHED option is used, any group of multiple blanks is reduced to a single blank for output of the record.

If the TRUNCATED option is used, a line too long for the terminal is truncated to fit on one terminal line.

If the PAGEFORMAT option is specified, the output appears in a format similar to the page mode output. That is, the full length of the sequence number field is displayed, immediately followed by the contents of the text field (the blank or asterisk flag character is omitted).

### Examples

```
L
100 THIS IS IN BOTH FILES
200 DIFFERENT COLUMNS
300 SAME COLUMNS
400 NEW FILE
#

LIST A/B
#FILE (UZER)A/B ON USERPACK
100 THIS IS IN BOTH FILES
150 OLD FILE
200 DIFFERENT COLUMNS
300 SAME COLUMNS
#
```

## MATCH Command (cont.)

---

MATCH A/B:T,C  
#UPDATING  
#MATCHING A/B TO WORKFILE

-150 OLD FILE

-200 DIFFERENT COLUMNS  
R200 DIFFERENT COLUMNS

300 SAME COLUMNS  
I400 NEW FILE  
#

MATCH A/B TO WORKFILE @15-20  
#MATCHING A/B TO WORKFILE  
-150,R200,I400

# MCS

## Syntax

— MCS —<MCS name>—————|

<MCS name>

—<file title>—————|

## Explanation

The MCS command causes control of the station to be transferred to another MCS. The <MCS name> must be declared in NDII for the current data comm system. If an unsaved work file exists, the following message is displayed:

```
#REMOVE OR SAVE WORKFILE
```

The user must either save or remove the work file and re-enter the MCS command before the station can be transferred.

The user is logged off and the session is terminated before control of the station is given to the desired MCS.

This command is the same as the ?MCS control command except that this form of the command will be queued if the station is busy, and the control command will not. Also, the user must be logged on to use this form.

## Examples

```
MCS SYSTEM/DIAGNOSTICMCS ON UZERPACK
#END SESSION 4241 ET=58:46.2 PT =1.8 IO=0.3
#USER = UZER 10:07:55 04/21/78
SYSTEM/DIAGNOSTICMCS(3.0.0) NIF:ELMONTENDL(2.9.0) YOU ARE T1(LSN:1)
```

```
MCS SYSTEM/TIOMLS ON PACK
#END SESSION 4245 ET=24:30.0 PT=0.7 IO=0.1
#USER = UZER 10:35:50 04/21/78
[ TIOMCS ] == WELCOME == YOU ARE TTY0 (LSN 13) SESSION 4247
```

# MERGE

### Syntax

```
-- MERGE <file title> <sequence range list> <file title>
```

### Explanation

The MERGE command copies a file into the work file, collating the sequence numbers from both files and preserving the work file records wherever matching sequence numbers occur. A number sign (#) acknowledges completion of the merge.

The < file title > construct specifies the file to be copied into the work file and can be any file to which the user is allowed access.

The < sequence range list > option limits the merge to specific portions of the file. The default is the entire file.

The UPDATE command is implicitly invoked after the MERGE has been executed.

The MERGE command performs the same function as RMERGE except that where sequence numbers coincide, the work file version, rather than the < file title > version, takes precedence.

### Example

```
LIST
100 WORKFILE - LINE 1
200 WORKFILE - LINE 2
300 WORKFILE - LINE 3
400 WORKFILE - LINE 4
#
```

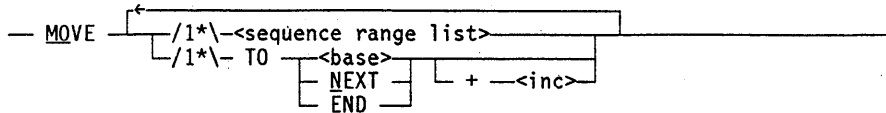
```
LIST MERGEFILE
#FILE (USERID)MERGEFILE ON DISK
100 MERGEFILE - LINE 1
150 MERGEFILE - LINE 2
200 MERGEFILE - LINE 3
250 MERGEFILE - LINE 4
#
```

MERGE MERGEFILE  
#UPDATING  
#

LIST  
100 WORKFILE - LINE 1  
150 MERGEFILE - LINE 2  
200 WORKFILE - LINE 2  
250 MERGEFILE - LINE 4  
300 WORKFILE - LINE 3  
400 WORKFILE - LINE 4  
#

# MOVE

### Syntax



### Explanation

The MOVE command moves lines from one place to another within the work file. The lines are moved as a contiguous block and are assigned new sequence numbers. Lines are moved, not copied; lines in the <sequence range list> no longer reside at their old numbers. (To copy lines in the work file, use the INSERT command.)

The <sequence range list> construct specifies the sequence range or ranges of the lines to be moved within the work file.

The new sequence numbers of moved lines are determined by assigning an initial value to the first line and incrementing that value for each succeeding line. The initial value is specified in one of three ways:

- The <base> option specifies the value explicitly. The <base> value must be an <integer>.
- The NEXT option sets the initial value to the next sequence number that would have resulted from the most recent MOVE, INSERT, RESEQ, or SEQ command. A default value of 100 is used if no such command has appeared since the last MAKE, GET, or DELETE ALL command.
- The END option sets the initial value to the largest sequence number currently in the file, plus the currently specified (or default) increment.

The <inc> construct defines the increment value to use for successive sequence numbers in the block. The increment value must be an <integer>. If no increment value is specified, the increment value from the most recent MOVE, INSERT, RESEQ, or SEQ command is used (100 is used if none of these commands have been used since the last MAKE, GET, or DELETE ALL command).

The range of new sequence numbers cannot overlap any lines already in the file, nor can the numbers exceed the largest sequence number that can be expressed in the sequence number field. If either of these conditions occurs, an error message will be displayed and the move will not be done.

The UPDATE command is implicitly invoked after the MOVE has been executed.

**Examples**

```
GET TESTONLY;LIST
#WORKFILE TESTONLY: SEQ, 15 RECORDS, SAVED
100 LINE 1
200 LINE 2
300 LINE 3
400 LINE 4
500 LINE 5
600 LINE 6
700 LINE 7
800 LINE 8
900 LINE 9
1000 LINE 10
1100 LINE 11
1200 LINE 12
1300 LINE 13
1400 LINE 14
1500 LINE 15
#
```

```
MOVE 1000-1300 TO 10+1
#UPDATING
#
```

```
L
10 LINE 10
11 LINE 11
12 LINE 12
13 LINE 13
100 LINE 1
200 LINE 2
300 LINE 3
400 LINE 4
500 LINE 5
600 LINE 6
700 LINE 7
800 LINE 8
900 LINE 9
1400 LINE 14
1500 LINE 15
#
```

## MOVE Command (cont.)

---

MOVE 1400,1500 TO NEXT

#UPDATING

#

L

10 LINE 10

11 LINE 11

12 LINE 12

13 LINE 13

14 LINE 14

15 LINE 15

100 LINE 1

200 LINE 2

300 LINE 3

400 LINE 4

500 LINE 5

600 LINE 6

700 LINE 7

800 LINE 8

900 LINE 9

#



## NEWS

### Syntax

— NEWS —————|

### Explanation

The NEWS command causes the contents of the CANDE news file to be listed on the user's terminal. If no news file exists, CANDE responds with the following display:

```
#NO NEWS
```

### Example

```
NEWS  
COMING UP ON NEW RELEASE NEXT WEEK  
#
```

# NEXT

### Syntax

— NEXT 

+
-

 <integer>

### Explanation

The NEXT command initiates page mode and allows movement through the work file while in page mode. By transmitting NEXT on the CANDE command line, the user can invoke page mode and display the first page of records in the work file. Once in page mode, the user can transmit *NEXT + <integer>* or *NEXT - <integer>* to shift <integer> pages through the work file forward or backward from the displayed page. The cursor should be positioned immediately after the last character of the NEXT command when the command is transmitted. However, the TAB key can be used to advance the cursor over the next command. If this results in the cursor being positioned beyond the first period (.) of the column indicator line, CANDE ignores the period. The user can transmit *NEXT +* or *NEXT -* from the first column of any line on the displayed page, with the same result.

When a non-page-invoking CANDE command is transmitted, page mode is temporarily interrupted. The user can transmit NEXT to return to page mode, which displays a page the specified number of pages (<integer>) away from the page that was displayed immediately before page mode was interrupted.

The term *page* used here is synonymous with *screenful*. The size of the page displayed on the screen is specified by using the TERMINAL command. (For more information, refer to the TERMINAL command.) *NEXT + <integer>* causes the file to be shifted forward by <integer> multiplied by the number of records in each page. After the shifting is done, a page of records is displayed. If less than a page is left in the file, the remaining lines are displayed along with a line providing the following message:

```
#DISPLAY COMPLETE
```

If <integer> is not specified, 1 is assumed. If + or - is not specified, + is assumed.

The token NEXT in the upper left-hand corner of the screen indicates that the user is in page mode. CANDE ignores other tokens on the top line when the user transmits page-mode input. CANDE displays NEXT + if *NEXT +* was just completed and NEXT - if *NEXT -* was just completed. This information is for the user's information. CANDE ignores this when retransmitted unless the user indicates it is a command by placing the cursor immediately after *NEXT +* or *NEXT -* and transmitting.

If the user transmits with the cursor in the home position (the upper left-hand corner of the screen) or anywhere other than immediately after *NEXT +* or *NEXT -*, CANDE processes the entire input and displays the next page forward, regardless of whether *NEXT +* or *NEXT -* appears at the top of the screen. Refer to "Page Mode Operations" in Section 1, "General Information," for more information.

If a terminal is incapable of page mode, the LIST command should be used to display records in amounts that the terminal is capable of accepting.

**Examples**

The following examples will display the fifth page forward from the page currently being displayed:

```
NEXT 5  
N +5
```

The following example will display the page previous to the page currently being displayed:

```
NEXT -
```

The following example will display the page that follows the currently displayed page:

```
N
```

# PAGE

### Syntax

— PAGE <sequence range>

### Explanation

The PAGE command initiates page mode and allows movement through the work file. By transmitting PAGE from the CANDE command line, the user can invoke page mode and display the first page (screenful) of records in the work file. However, if the terminal can display only one or two lines, this command and page mode cannot be used. (Refer to “Page Mode Operations” in Section 1, “General Information.”) If a terminal is not capable of page mode, the LIST command should be used to display records in amounts that the terminal is capable of accepting.

### PAGE

PAGE <sequence range>

PAGE displays a page of records beginning with a specified sequence number. The default value for <sequence range> is the first record through the last record in the file. If only one sequence number of the <sequence range> is specified, the sequence range requested is that sequence number through the last record in the file.

If a first and a last sequence number are specified as the <sequence range>, CANDE begins a page with the first sequence number in the range. If the first sequence number specified in a <sequence range> does not exist, the page begins with the next higher existing sequence number. If the last sequence number in the range is not reached by the end of the page, NEXT+ can be specified to allow paging through the file until the last sequence number in the range is reached. If any command other than NEXT+ is specified, CANDE discards the knowledge of the last sequence number in the range and allows the user to proceed to the last record in the file. For more information, refer to the NEXT command.

If *PAGE END* is entered, the last page of the work file is displayed.

Note that the meaning of a single sequence number as a <sequence range> for LIST is different from that for PAGE. *LIST <sequence number>* displays the single record at the specified sequence number, while *PAGE <sequence number>* displays a page of records, the first record of which is the specified sequence number.

**Examples**

The following command displays a full page of records that begins with the first record of the file:

PAGE

The following example will display records that are in the file beginning at sequence number 1000 and continuing until the page is filled or the sequence numbers reach 2000. If 2000 is not reached on the first displayed page, subsequent NEXT+ commands will display pages until the sequence number reaches 2000.

PA 1000-2000



If a usercode does not have a password associated with it, this command cannot be used. A user can assign a password to a usercode that has none by using the password when logging on. (The maximum password value in the USERDATAFILE must be greater than zero.) This newly assigned password can then be used as the < old password > in the password command. (Refer to "User Identification and Logging On" in Section 1, "General Information," for further information.)

In each of the examples below, FIRSTONE is the first password in the password list (or its successor if FIRSTONE was explicitly changed, as is done in the last example shown) for that usercode.

To add PW to the password list, the following command is entered:

```
PASSWORD FIRSTONE + PW PW
```

To delete PW from the password list, the following command is entered:

```
PASSWORD FIRSTONE - PW PW
```

To replace the whole password list with the single entry PW, the following command is entered:

```
PASSWORD FIRSTONE = PW PW
```

To remove the requirement for a password at log on, the following command is entered:

```
PASSWORD FIRSTONE - FIRSTONE FIRSTONE
```

To replace the password OLDPW by PW, the following command is entered:

```
PASSWORD OLDPW PW PW
```

The operations for password lists are constrained by the MINPW and MAXPW bounds established by the installation (in the USERDATAFILE) for the number of passwords that can be defined for any given usercode.

The PASSWORD command alters the USERDATAFILE. Occasionally, the USERDATAFILE is frozen so that no changes can be made. If any attempt to change a password (or password list) is made during this time, CANDE rejects the attempt with the following message:

```
#USERDATAFILE FROZEN; TRY LATER.
```

## PASSWORD Command (cont.)

---

### Example

```
PASSWORD
#ENTER CURRENT PASSWORD, PLEASE.
ROBERT
#ENTER NEW PASSWORD PLEASE.
ROBBY
#RE-ENTER NEW PASSWORD PLEASE.
ROBBY
#
```





When PDEF is the only input entered, CANDE reports the current value of the PRINTDEFAULTS attributes.

The asterisk (\*) causes the current value of the PRINTDEFAULTS attributes to return to the default of the usercode associated with the session.

The period (.) causes the session PRINTDEFAULTS attributes to be set to null. In this case, PRINTDEFAULTS attributes are not supplied by CANDE on any task execution request.

If the PRINTDEFAULTS attributes are explicitly changed during a session, those attributes remain in effect until the session is terminated or until they are changed by another PRINTDEFAULTS action.

If the PRINTDEFAULTS attribute DESTINATION is specified for the user's session, and the CANDE command DESTNAME is issued with a station name, the PRINTDEFAULTS DESTINATION value takes precedence. The following message is displayed when either DESTNAME or PRINTDEFAULTS DESTINATION is specified or changed and the other is already specified:

WARNING: PRINTDEFAULTS DESTINATION OVERRIDES DESTNAME.

This warning is also displayed at CANDE log-on time if both CANDEDESTNAME and PRINTDEFAULTS are specified for the user in the USERDATAFILE.

When the PRINTDEFAULTS attributes are specified in a task-initiating statement (for example, COMPILE, EXECUTE, RUN, and UTILITY), the new value overrides the former value for the session for that request only. No abbreviation for PRINTDEFAULTS is accepted when it is used as a task-initiating command. (Refer to the COMPILE, EXECUTE, RUN, and UTILITY statements for the correct syntax.)

### Examples

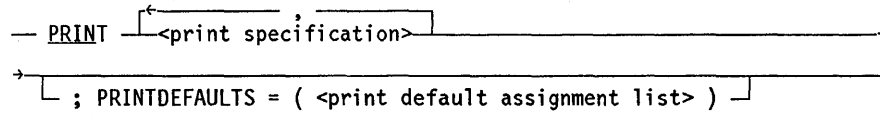
PDEF.

PDEF = (BANNER=TRUE, NOTE="HEY THERE!")

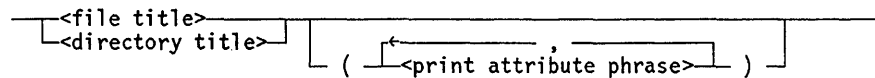
PDEF\*

# PRINT

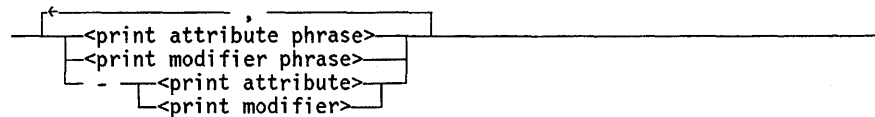
## Syntax



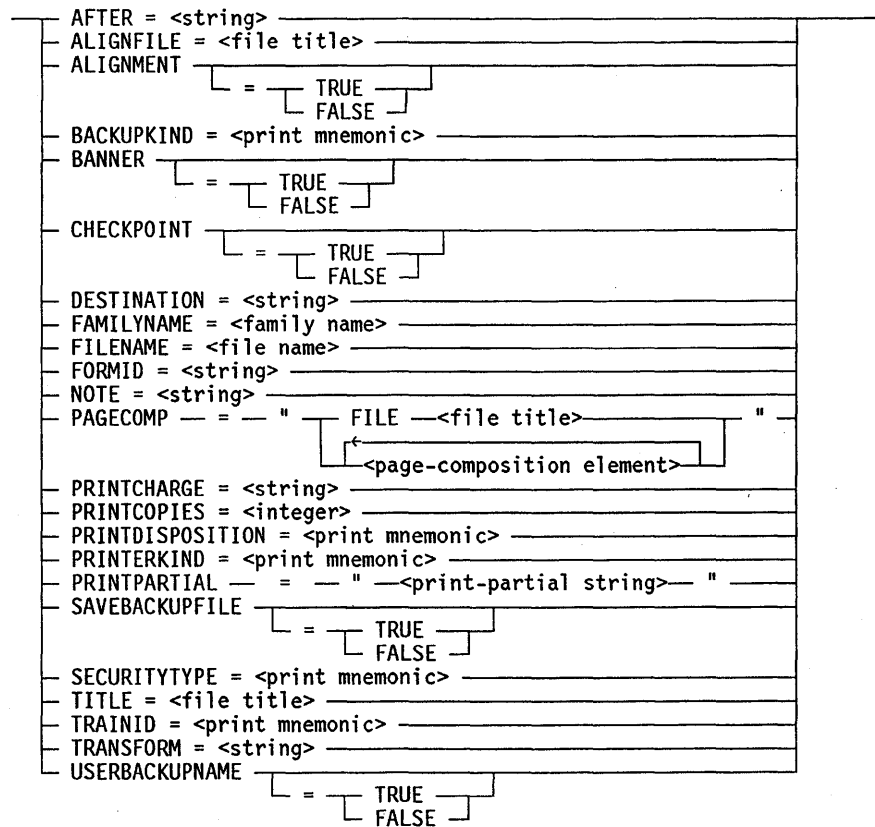
### <print specification>



### <print default assignment list>

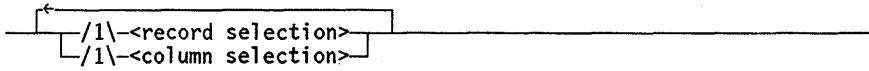


### <print attribute phrase>

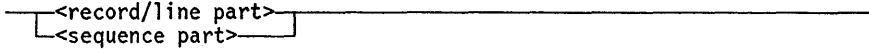


# PRINT Command (cont.)

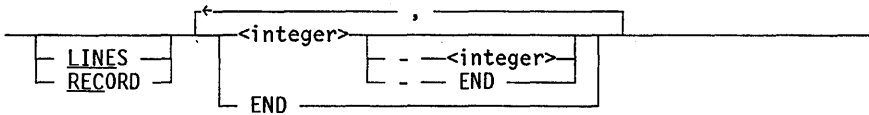
## <print-partial string>



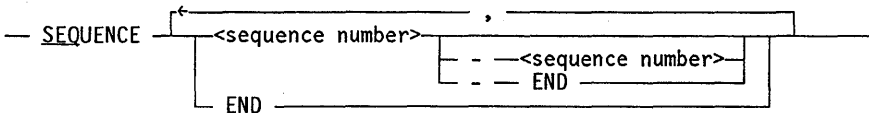
## <record selection>



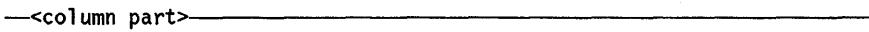
## <record/line part>



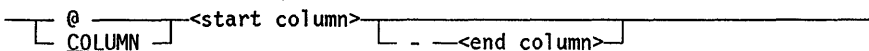
## <sequence part>



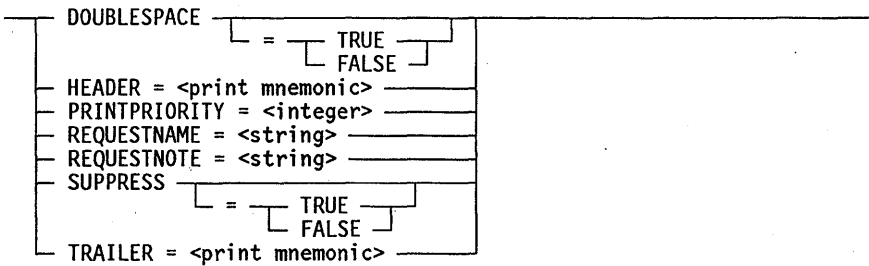
## <column selection>



## <column part>



## <print modifier phrase>



## <print mnemonic>

One of the mnemonic values defined in the *File Attributes Reference Manual* for a particular print-related file attribute, or one of the mnemonic values defined in the *Print System Guide* for a particular print modifier, as shown in Tables 3-6 and 3-7.

**Table 3-6. Print Attributes and Corresponding Mnemonic Values**

<b>Print Attribute</b>	<b>Mnemonic Values</b>
BACKUPKIND	DONTCARE, DISK, PACK, TAPE, TAPE7, TAPE9, TAPEPE
PRINTDISPOSITION	CLOSE, DIRECT, DONTPRINT, EOJ, EOT
PRINTERKIND	APT, DONTCARE, D630, D630E, EPSONFX, EPSONLQ, EXPRESS, HPGL, IMAGEPRINTER, KANJIIPRINTER, LINEPRINTER, PCL3, PCL4, PCL5, PLOTTER, POSTSCRIPT, PROPRINTER, PROPRINTERXL, TTY
SECURITYTYPE	CONTROLLED, GUARDED, PRIVATE, PUBLIC
TRAINID	ASCII64A, ASCII64B, ASCII72, ASCII96A, ASCII96B, BCL64, BRAZIL, B300B500, DENMARK, DENMARKNORWAY, EBCDIC18, EBCDIC48, EBCDIC64A, EBCDIC64B, EBCDIC72, EBCDIC96, FORTRAN48, GERMANYAUSTRIA, ITALY, KATAKANA, LATINPORTUGAL, LATINSPAIN2, LATINSPAIN3, NOID, OCRAALPHANUMERIC, OCRANUMERIC, OCRBNUMERIC, RPG48, SWEDENFINLAND2, SWEDENFINLAND3, SWEDENOCRB, TURKEY, UK, UK3500, UK6500, YUGOSLAVIA

**Table 3-7. Print Modifiers and Corresponding Mnemonic Values**

<b>Print Modifier</b>	<b>Mnemonic Values</b>
HEADER	SUPPRESSED, UNCONDITIONAL
TRAILER	CONDITIONAL, SUPPRESSED, UNCONDITIONAL

For a description of all the print attributes and the print modifiers, refer to the *Print System Guide*.

### Explanation

CANDE recognizes the PRINT command as a Work Flow Language (WFL) command; that is, CANDE passes the command to WFL for processing. WFL then passes a PRINT request to the Print System (PrintS/Reprints) for asynchronous processing. For additional information, refer to the *WFL Reference Manual* and the *Print System Guide*.

*Note:* Because CANDE can run with different release versions of the MCP and other software, it is possible that a feature of software might not work as described in this Mark release version of the manual. For example, if the Mark 4.0 version of CANDE is run with the Mark 3.9 version of the MCP and Print System (PrintS/Reprints) software, then a specification of a Mark 4.0 print attribute in the PRINTDEFAULTS assignment, as documented in the Mark 4.0 release version of this manual, might cause an error or unexpected result.

### <print specification>

The print specifications in the PRINT statement specify the files to be printed or punched. These files must be printer or punch backup files. Each print specification can include its own print attribute phrases, which affect only the specified file or directory.

### <print attribute phrase>

Various print attribute phrases can be included in the PRINT statement to specify print-related file attributes that control the creation, routing, and formatting of backup files. These file attributes are described in the *Print System Guide* and the *File Attributes Reference Manual*.

### <print modifier phrase>

A print modifier phrase can be included in a PRINT statement to specify additional requirements for the processing of a print request. Print modifiers can be used with a PRINT statement only through the PRINTDEFAULTS task attribute. For more information about print modifiers, see the *Print System Guide*.

### PRINTDEFAULTS Option and <print default assignment list>

A PRINTDEFAULTS specification can be included at the end of a PRINT statement to provide a new set of default values for some print-related file attributes and print modifiers. These values replace defaults that were inherited from the job or from the system.

The print attribute phrases and print modifier phrases that appear in the print default assignment list are merged into the current print defaults. The - <print modifier> and - <print attribute> forms reestablish the system default value for that print modifier or print-related file attribute.

If a print specification and the PRINTDEFAULTS specification both assign values to the same print-related file attribute, then the value assigned in the print specification takes precedence. For details, see the *Print System Guide*.

#### File Attributes of <print attribute phrase>

The printer-related file attributes of the print attribute phrase are described in the *File Attributes Reference Manual* and the *Print System Guide*. A few of these printer-related file attributes are described in the following paragraphs.

#### PAGECOMP File Attribute

The PAGECOMP file attribute is used to specify a combination of page-composition elements to make full use of a variety of remote printer capabilities. Among those printer capabilities that are supported by the PAGECOMP file attribute are the Hewlett-Packard Printer Command Language (PCL) and the PostScript® page-description languages.

Refer to the *Print System Guide* for more information about the types of printers that are supported and the printing capabilities that can be specified through the PAGECOMP file attribute.

#### PRINTPARTIAL File Attribute

The PRINTPARTIAL file attribute is used to specify the lines, records, or sequence numbers of a file to print. Columns of the file can also be specified. The PRINTPARTIAL file attribute enables you to print part of a file rather than the entire file. The constructs that are used to describe the PRINTPARTIAL file attribute option are <print-partial string>, <record selection>, <column selection>, <record/line part>, <sequence part>, and <column part>.

The print-partial string specifies the portion of a file to be printed. The print-partial string must be enclosed within quotation marks (" ").

You can specify a single line or record, or a range of lines or records, of a file to be printed. When specifying either lines or records, note that records are relative to 0 (zero) and lines are relative to 1; therefore, record 0 (zero) is equivalent to line 1. LINES is the default if neither RECORD nor LINES is specified in the <record/line part> construct.

A sequence line-number or a range of sequence line-numbers can be specified for nonbackup file types, such as symbol files or sequential files. Because backup files do not have sequence numbers, the SEQUENCE clause is not meaningful for backup files. However, if the SEQUENCE clause is used with a backup file, a sequencing of line numbers is begun with 100 and incremented by 100, in the same way that CANDE currently does the numbering of new nonbackup files. For example, if the lines of a backup file are numbered from 1 through 10 and the PRINTPARTIAL option of the

## PRINT Command (cont.)

---

PRINT command is specified, then the line numbers of the file are displayed as 100 through 1000, incremented by 100.

Columns of a file can also be specified and must be in the range from 1 through the MAXRECSIZE of the file. For nonbackup files, if a column number or range is not specified, the sequence numbers are placed before the print line. If a column range is specified for a nonbackup file, then the sequence numbers that would ordinarily appear before each line do not appear.

The range of lines, records, sequence numbers, and columns must be specified in ascending order.

For example, the following statement prints only columns 6 through 77 of lines 30 through 90 of a file named FILE1, which contains more than 90 lines:

```
PRINT FILE1 (PRINTPARTIAL = "LINES 30-90 COLUMN 6-77")
```

Refer to the *File Attributes Reference Manual* and the *Print System Guide* for more information about the PRINTPARTIAL file attribute.

### TRANSFORM file attribute

The TRANSFORM file attribute specifies a file transform, which is a general-purpose function that is applied to each line of a backup file between the time the file is created and the time it is printed. A file transform is applied to the file no matter what device prints it. A transform function associated with a device is called a device transform, which is a general-purpose function that processes all data sent to that device. A device transform is associated with a device through the use of the TRANSFORM option of the PS CONFIGURE system command. If a file transform is specified for a backup file that is routed to a printer with a device transform, the file transform is applied first, and then the device transform is applied.

### Examples

In the following examples, the percent sign (%) indicates the defined continuation character; refer to the CONTINUE command for more information about the continuation character.

The following are simple PRINT statements that cause the specified files to be printed:

```
PRINT DRONE/CLONE;
```

```
PRINT (JOHNS)ADD/BACK ON THREEPACK, (CAY)INVENTORY/LIST;
```

The following PRINT statements contain a print attribute phrase as part of the print specification:



```
PRINT (WENDY)FREE/CODE (BANNER = TRUE,%  
#%  
NOTE="Review printout for Bill Wyman");  
  
PRINT (JAKE)SCRAG/EXTRAS (SAVEBACKUPFILE = TRUE,%  
#%  
PRINTERKIND = LINEPRINTER);
```

The following example includes individual print specifications and common print modifiers in a PRINTDEFAULTS specification:

```
PRINT (LIZA)CAVE/DEPTHS (DESTINATION="LP5",PRINTCHARGE=4328),%  
#%  
(GEORGE)WATER/COMP (DESTINATION="IP7",PRINTCOPIES=3);%  
#%  
PRINTDEFAULTS=(HEADER=SUPPRESSED,TRAILER=UNCONDITIONAL);
```

The following PRINT statements show a variety of PRINTPARTIAL option specifications.

```
PRINT (ACCT)LEDGER/AUGUST (PRINTPARTIAL="LINES 20-35")  
  
PRINT (ACCT)LEDGER/AUGUST (PRINTPARTIAL="RECORDS 19-34")  
  
PRINT (ACCT)LEDGER/3RDQTR (PRINTPARTIAL="LINES 10, 20, 30-END")  
  
PRINT (ACCT)LEDGER/3RDQTR (PRINTPARTIAL="SEQUENCE 2300,2900,4000-END")  
  
PRINT (ACCT)LEDGER/3RDQTR (PRINTPARTIAL="SEQ 100-900 COLUMN 1-72")  
  
PRINT (ACCT)LEDGER/3RDQTR (PRINTPARTIAL="SEQ 1000-END @ 40-72")
```

The following PRINT statements illustrate typical PAGECOMP strings for a remote printer. Remember that frequently needed PAGECOMP strings can be stored in ordinary DATA or SEQ disk files.

The following example prints a file in portrait orientation.

```
PRINT EXEC/SUMMARY (PAGECOMP="PORTRAIT CPL=72 LPP=60 LM=0.8  
FONT DEFAULT=HELVETICA(10 PT)")
```

The following example prints a file in landscape orientation.

```
PRINT ANNUAL/BUDGET (PAGECOMP="LANDSCAPE CPL=132 LPP=66")
```

The following example prints an unusually wide listing in landscape orientation at maximum print density on an AP9415 printer.

```
PRINT JUMBO/SPREADSHEET (PAGECOMP="LANDSCAPE CPL=315 LPP=144")
```

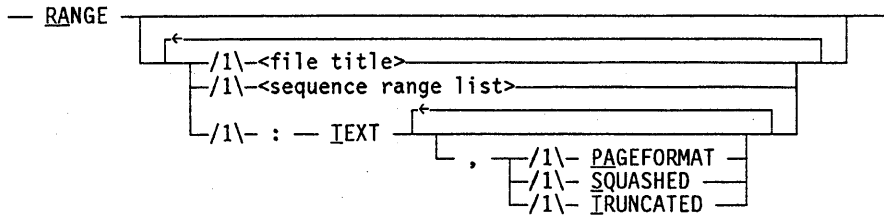


## **PRINTDEFAULTS**

PRINTDEFAULTS is another name for the PDEF command. (Refer to the PDEF command.)

## RANGE

### Syntax



### Explanation

The RANGE command displays the number of lines within one or more sequence ranges of a file (by default, the work file). The lowest and highest sequence numbers in the sequence range or ranges are also displayed.

The <file title> construct specifies the file for which the range information is to be displayed; this file can be any file to which the user is allowed access. If unspecified, the work file is assumed. In this case, an update is done before the RANGE command is invoked.

The <sequence range list> construct limits the ranging to particular areas of the file; the default is the entire file. If a <sequence range> in the <sequence range list> consists of a single sequence number or the single keyword END, the sequence numbers (and optionally, the text) of the preceding and succeeding lines are also displayed.

The TEXT option causes entire lines to be printed in the range output, instead of sequence numbers only. When the TEXT option is used and the PAGEFORMAT option is not specified, all lines that have been marked with the current MARKID will have an asterisk between the sequence number and the rest of the line. For more information on the MARKID, refer to the GET command, the MAKE command, and the MARKID command.

The three output format options available are PAGEFORMAT, SQUASHED, and TRUNCATED. These options affect the text as described below.

When the PAGEFORMAT option is specified, the output appears in a format similar to the page mode output. That is, the full length of the sequence number field is displayed, immediately followed by the contents of the text field (the blank or asterisk flag character is omitted).

When SQUASHED is specified, any group of multiple blanks is reduced to a single blank.

When TRUNCATED is specified, the record is truncated if the text portion of a record does not fit in the available line width of the terminal.

**Examples**

```
LIST
100 one
200 two
300 three
400 four
500 five
600 six
700 seven
800 eight
900 nine
1000 ten
#
```

```
RANGE
# 10 RECORDS: 100 THRU 1000
#
```

```
RANGE:TEXT,PA
# 10 RECORDS:
00000100one
  THRU
00001000ten
#
```

```
RANGE 500
400, 500, 600
#
```

```
RANGE 500:TEXT
400 four
500 five
600 six
#
```

```
RA 100-300,400-END,1025-1050
# 3 RECORDS: 100 THRU 300

# 7 RECORDS: 400 THRU 1000

#NO RECORDS IN 1025-1050
#
```

```
RA 550:TEXT
500 five
600 six
#
```

## RANGE Command (cont.)

---

The following example shows the RANGE command used with the TEXT option when some of the lines in the file have been marked with the current MARKID:

```
RA 550:TEXT
500*five
600*six
#
```

# RECOVER

## Syntax

```

— RECOVER —————|
| <recovery number> |
| REPEAT —————|
| : — ALL —————|
| : — BRIEF —————|

```

## Explanation

The RECOVER command processes a recovery file that, depending on the setting of the AUTORECOVER option of the USERDATAFILE file attribute, can recall the previous work file and restart the last task-type command such as RUN, UTILITY, COMPILE, and so on. The user can recover a desired recovery file by specifying a recovery number.

When a recovery file is recovered, the FAMILY, LANGUAGE, CONVENTION, and PRINTDEFAULTS file attributes are restored to those in effect at the time the work file was saved. The last command that was processed before the creation of the recovery file is restored as the current command. That is, the command is displayed in response to a ?SH command.

If the automatic recovery is enabled, the last task-type command that was processing but abnormally terminated is queued for execution after the recovery process is complete. If the automatic recovery is not enabled and the REPEAT option is specified, then the command (regardless of its type) is queued for execution after the recovery process is complete.

If a recovery number is not specified with the RECOVER command and the AUTORECOVER option is set to TRUE for the session, then CANDE lists the recovery file in the following format:

```
#RECOVERY DATA:
```

```

<recovery number> <work file name> ( <date> )
  [station name]
  [last command]

```

(Additional recovery files are listed in this format)

When RECOVER is specified with the :ALL option, the recovery files are listed in the same format in the previous example, regardless of the setting of the AUTORECOVER option.

If a recovery number is not specified with the RECOVER command and the AUTORECOVER option is reset to FALSE for the session, then CANDE lists the recovery file in the following format:

## RECOVER Command (cont.)

---

#RECOVERY DATA:

<recovery number> <work file name> ( <date> )

(Additional recovery files are listed in this format)

When RECOVER is specified with the :BRIEF option, the recovery files are listed in the same format in the previous example regardless of the setting of the AUTORECOVER option.

For more information on recovery numbers, refer to the discussion of recovery files in Section 1, "General Information."

### Examples

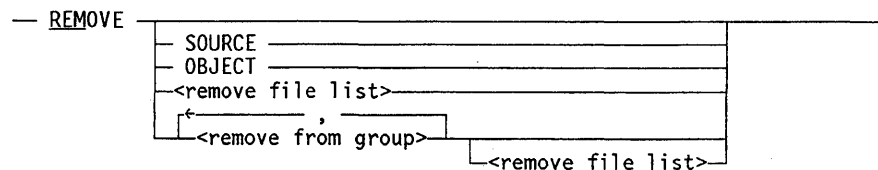
```
REC:B
#RECOVERY DATA ON USERPACK
 100 PAYABLE (04/21/91)
#
```

```
recover 100
#WORKFILE PAYABLE: SEQ
```

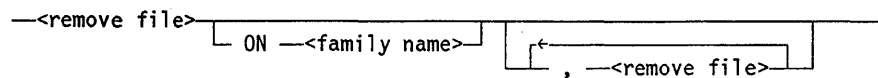


# REMOVE

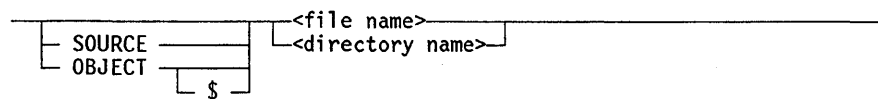
## Syntax



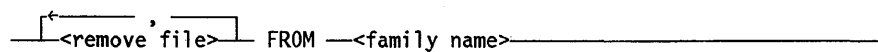
### <remove file list>



### <remove file>



### <remove from group>



## Explanation

The REMOVE command removes the work file, files, or directories from disk.

When a file is removed, it is immediately removed from the system directory; therefore, no program can find it. The actual data areas remain available to any program that has the file open at the time of the remove. CANDE saves a copy of the file in the work file area if the file being removed is required for the present work file. In this case, CANDE returns a message stating that the worksource has been retrieved. However, this protective action only occurs when the CANDE REMOVE command is used in the current session. If the work file source is removed by a program (or a WFL REMOVE or a CANDE REMOVE in another session), a copy will not be preserved.

If a <family name> construct is specified, the command applies only to the file name or directory name on the volume associated with the family name. For example, REMOVE FILEX ON DISK affects only files found on DISK even though the family specification for the session is FAMILY DISK = USERPACK OTHERWISE OTHERPACK.

If a family name is not specified, the command applies to the file name or directory name found on DISK. If DISK is also the <target family> in the family specification for the session, then the only files affected by the command are those on the <substitute family>. Files on the <alternate family> are not affected.

In a <remove from group> construct, a FROM <family name> clause applies to all the file names or directory names in that <remove from group>. An ON <family name>

clause applies only to the immediately preceding <file name>. An ON clause cannot precede a FROM clause in the same command. An ON <family name> clause can be specified only for the first name in a <remove file list> construct.

A work file source is not removed when specified with an ON <family name> or FROM <family name> clause if the family name is other than the substitute family; however, the work file source is removed when the work file is removed or saved.

When specifying a file name or directory name, a user logged on under a nonprivileged usercode cannot enter a usercode specification other than the usercode for the current session.

If the REMOVE command is used to remove a file whose LOCKEDFILE file attribute is set to TRUE, that file is not deleted. CANDE displays the following message to indicate that the file was not removed:

```
<file name> NOT REMOVED (LOCKEDFILE).
```

See the ALTER command earlier in this section for more information about changing the LOCKEDFILE attribute. For more information about the LOCKEDFILE file attribute, refer to the *File Attributes Reference Manual*.

If SOURCE or OBJECT is used as a file name, it must be enclosed in quotation marks to distinguish it from the keywords SOURCE and OBJECT.

The following paragraphs contain an explanation for each syntactic variation of the REMOVE command.

**REMOVE**  
**REMOVE SOURCE**  
**REMOVE OBJECT**

REMOVE removes the work file and any associated workobject file. If REMOVE SOURCE is used, the worksource file (source) is removed, but any associated workobject file (object) remains available to be executed or saved. If REMOVE OBJECT is used, the workobject file is removed, and the worksource file remains.

**REMOVE <file name>**  
**REMOVE SOURCE <file name>**  
**REMOVE OBJECT <file name>**  
**REMOVE OBJECT \$ <file name>**

REMOVE <file name> removes any file with the specified file name. Any object file associated with the file name is also removed. The REMOVE SOURCE <file name> form removes the file with the specified file name, but does not remove the object file associated with the file name. The REMOVE OBJECT <file name> form removes only the object file associated with the file name. If a dollar sign (\$) follows OBJECT, it indicates that the object file to be removed is stored as the file name, not as OBJECT/<file name>, so only the file with the specified file name is removed.

```
REMOVE <directory name>
REMOVE SOURCE <directory name>
REMOVE OBJECT <directory name>
REMOVE OBJECT $ <directory name>
```

REMOVE <directory name> removes all files in the directory with the specified directory name. If the files in the object directory associated with the directory name exist, they are also removed. The REMOVE SOURCE <directory name> form removes the files in the directory with the specified directory name but does not remove the files in the object directory associated with the directory name. The REMOVE OBJECT <directory name> form removes the files of the object directory associated with the directory name. If a dollar sign (\$) follows OBJECT, it indicates that the object files to be removed are stored in the directory with the specified directory name, not in the associated object directory, so only the directory with the specified directory name is removed.

### Examples

```
G D/F
#WORKFILE D/F: SEQ, 1 RECORD, SAVED
```

```
FIX 100/2/1
#
```

```
REMOVE D/=, F/=, TEMP, SOURCE UTIL, Z, OBJECT O/PROG
#WORKSOURCE D/F RETRIEVED
#1 FILE IN (UZER)D/= REMOVED ON USERPACK
#2 FILES IN (UZER)F/= REMOVED ON USERPACK
#1 FILE IN (UZER)OBJECT/F/= REMOVED ON USERPACK
#(UZER)TEMP ON USERPACK REMOVED
#(UZER)UTIL ON USERPACK REMOVED
#(UZER)OBJECT/Z ON USERPACK REMOVED (NO SOURCE)
#(UZER)OBJECT/O/PROG REMOVED ON USERPACK
```

```
FILES D
#NO FILE(S) ON USERPACK
```

```
WHAT
#WORKFILE D/F: SEQ, 1 RECORD (THRU 100)
```

```
REMOVE
#
```

```
REMOVE F, D/= FROM OTHERPACK
#(UZER)F REMOVED ON OTHERPACK
#3 FILES IN (UZER)D/= REMOVED ON PACK
```

# RENEW

### Syntax

— RENEW —<sequence range list>—————|

### Explanation

The RENEW command allows portions of the work file to be renewed to the condition in which they existed the last time the work file was updated. The sequence ranges specified need not be in ascending order.

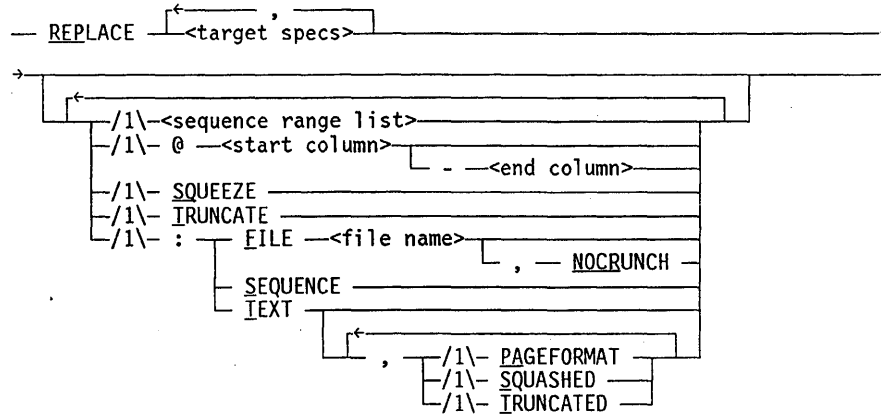
### Example

```
GET X
#WORKFILE X: SEQ, 3 RECORDS, SAVED
LIST
2000 THE RAIN IN SPAIN STAYS  MAINLY
4000 IN THE PLAIN.
6000 THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.
#
FIX 4000/IN/ON
#
8000 THE EARLY BIRD GETS THE WORM..
DELETE 6000
#
LIST
2000 THE RAIN IN SPAIN STAYS  MAINLY
4000 ON THE PLAIN.
8000 THE EARLY BIRD GETS THE WORM.
#

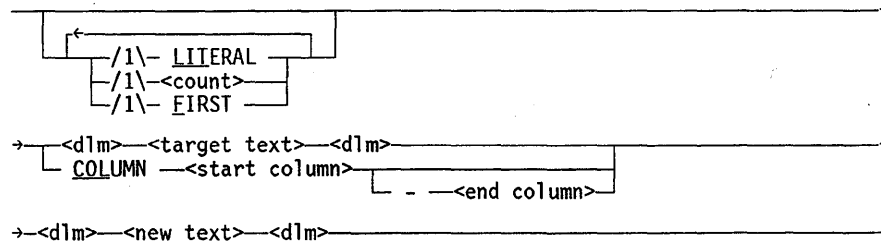
RENEW 0-10000
#
LIST
2000 THE RAIN IN SPAIN STAYS  MAINLY
4000 IN THE PLAIN.
6000 THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.
#
```

# REPLACE

## Syntax



## <target specs>



## Explanation

The REPLACE command searches all, or part, of the work file for occurrences of specific target text and replaces the target text with new text. The system responds with a number sign (#) to signify completion.

The UPDATE command is implicitly invoked after the REPLACE command has been executed.

### LITERAL

The LITERAL option specifies that the search mode for the target text is to be the literal mode. The default search mode is token mode. Refer to the FIND command.

### <count>

The <count> option, if specified, must be an <integer>. The value of <count> limits the maximum number of replacements to be performed. Any appearances of the target text after <count> occurrences are ignored.

## REPLACE Command (cont.)

---

### FIRST

If the FIRST option is specified, each line is searched only for the first occurrence of the target text. Any occurrences later in the line are ignored.

### <dlm>

The <dlm> character may be any special character except the comma (,), colon (:), or at sign (@).

The target text for a replace must be specified in one of the three forms described in the following paragraphs.

### <dlm> <target text> <dlm>

The <dlm> <target text> <dlm> form explicitly defines the target text, which must be bounded by two matching delimiters.

If a hexadecimal escape character is set through the CANDE command ESCAPE, then the target text can contain hexadecimal values delimited by the defined hexadecimal escape character. However, the target delimiter (<dlm>) cannot be the same as the hexadecimal escape character. For more information about the hexadecimal escape character, refer to the ESCAPE command earlier in this section.

### COLUMN <start column> - <end column>

The COLUMN <start column> - <end column> form defines the target text to be all text on each line between the character positions specified by <start column> and <end column>, inclusive.

The COLUMN <start column> form specifies that <new text> is to be inserted from the <start column> character position of each line. The remainder of each line is shifted right to accommodate the <new text>.

Adjustment for different lengths of target text and new text is made by shifting the right side of the line (or column range specified by @<start column> - <end column>) to the left or right, deleting or adding trailing blanks as required. A line overflow error is detected whenever the adjustment would shift nonblank characters off the end of the line (or the (@) column range). By default, the replacement is not performed if an overflow occurs, and a "# <sequence number> -SKIPPED." message is displayed.

### <sequence range list>

The <sequence range list> limits the REPLACE operation to one or more <sequence range>s in the work file.

@ <start column>  
@ <start column>--<end column>

When @ <start column> is specified, the REPLACE operation can only take place at the <start column> position. When @ <start column>--<end column> is specified, the REPLACE operation is limited to the specified column range. The entire column range is included in any left or right justification that may be necessary to adjust for different lengths of target and new text. All text outside this column or column range is not affected in any way by the replace or any text justification that may occur.

#### SQUEEZE

The SQUEEZE option causes strings of multiple blanks to be shortened (if possible) to make room for the substitution. If sufficient squeezing is impossible, the line is left *unsqueezed*, the substitution is skipped, and a "# <sequence number> -SKIPPED." message is displayed.

**REPLACE Command (cont.)**

---



### TRUNCATE

The TRUNCATE option causes the characters shifted off the right side of a line overflow to be lost.

### FILE <file name>

The FILE <file name> option causes each line modified by the REPLACE to be written to a file in the user's library. The <file name> specifies the title of that file. If a file entitled <file name> already exists in the user's library, an error message to that effect is displayed.

Files created by the REPLACE command are crunched by default. If an uncrunched file is desired, the output option NOCRUNCH can be specified.

### SEQUENCE

The SEQUENCE option causes the sequence number of each line modified by the REPLACE command to be displayed at the terminal. Asterisks in this output flag lines that had more than one replacement.

### TEXT

The TEXT option causes each line modified by the REPLACE to be displayed at the terminal in the modified form. When the TEXT option is used and the PAGEFORMAT option is not specified, all lines that have been marked with a current MARKID will have an asterisk between the sequence number and the rest of the line, unless the REPLACE command has replaced the record ID of the marked lines. For more information on the MARKID, refer to the GET, MAKE, and MARKID commands.

The PAGEFORMAT, SQUASHED, and TRUNCATED options apply to the terminal output only and do not affect the line images in the work file. When the PAGEFORMAT option is specified, the output appears in a format similar to the page mode output. That is, the full length of the sequence number field is displayed, immediately followed by the contents of the text field (the blank or asterisk flag character is omitted). The SQUASHED option causes a sequence of blanks to be printed as a single blank. The TRUNCATED option causes the output line to be truncated to the character width of the terminal, if necessary. By default, lines that exceed terminal width are split across two or more lines.

The REPLACE command is different from the FIX command. Refer to the FIX command for more information.

When @ <start column> or @ <start column>-<end column> is specified and the target text and new text are different lengths, the remaining unspecified columns of text are not affected by the replace or any text justification. The following example illustrates this feature:

## REPLACE Command (cont.)

---

```
G FRUIT
#WORKFILE FRUIT: SEQ, 1 RECORD, SAVED
```

```
LIST
100 WATERMELON****PLUM
#
```

```
REPLACE /WATERMELON/ /PEAR/ @1-10
#WORKFILE FRUIT
#UPDATING
#
```

```
LIST
100 PEAR      ****PLUM
```

If a different column range is specified for the same work file as before, the results are as indicated in the following example:

```
REPLACE /WATERMELON/ /PEAR/ @1-15
#WORKFILE FRUIT
#UPDATING
#
```

```
LIST
100 PEAR****P      LUM
```

The \*\*\*\*P is part of the specified column range, so it is included in the left justification.

### Examples

```
G FRUIT
#WORKFILE FRUIT: SEQ, 5 RECORDS, SAVED
```

```
LIST
100 APPLE ORANGE PEAR
200 GRAPE PLUM APPLE
300 CHERRY APPLE LEMON
400 LIME PEAR BANANA
500 APPLE ORANGE APPLE
#
```

```
REPLACE /APPLE/ /LEMON/ :S
#WORKFILE FRUIT
#UPDATING
100, 200, 300,*500
#
```

```
LIST
100 LEMON ORANGE PEAR
200 GRAPE PLUM LEMON
300 CHERRY LEMON LEMON
400 LIME PEAR BANANA
500 LEMON ORANGE LEMON
#
```

```
REPLACE .PEAR..PEACH.:FILE PEACHBOWL
#WORKFILE FRUIT
#UPDATING
#
```

```
LIST PEACHBOWL
100 LEMON ORANGE PEACH
400 LIME PEACH BANANA
```

The following example shows the REPLACE command used with the TEXT option when a current MARKID has been placed in some lines:

```
REPLACE .L..1.:TEXT
#WORKFILE FRUIT
#UPDATING
100 1EMON ORANGE PEACH
200*GRAPE PLUM 1EMON
300 CHERRY 1EMON 1EMON
400 1IME PEACH BANANA
500*1EMON ORANGE 1EMON
#
```

# RESEQ

### Syntax

```
— RESEQ [ <sequence range> ] [ <base> ] [ + <inc> ]  
         OVERRIDE
```

### Explanation

The RESEQ command assigns new sequence numbers to lines in the work file without changing the order of appearance of any lines.

An initial value is assigned as the sequence number of the first line to be resequenced. The value is incremented for each subsequent line.

If a <sequence range> is specified, only the specified part of the work file is resequenced. By default, the entire work file is resequenced.

An initial value can be specified as the integer <base>; otherwise, the starting value of the sequence range is used. An <inc> value can be specified as the increment for successive new sequence numbers. If no increment is specified, the increment from the most recent MOVE, INSERT, RESEQ, or SEQ command is used. The default of 100 is assumed if no such command has occurred since the last MAKE, GET, or DELETE ALL command.

If a <sequence range> is specified, the new sequence numbers must fall within the bounds of that <sequence range>. RESEQ does not change the order in which the lines appear.

The OVERRIDE option causes all data currently in the sequence field of the file to be ignored. This option can be used only with a complete file that has no pending changes. The expected application is GET <file title>; RESEQ OVERRIDE, thereby sequencing a file that was previously unacceptable to CANDE because of sequence errors or nondigits in the sequence field.

The UPDATE command is implicitly invoked after the RESEQ command has been executed.

**Examples**

```
list
5 FIRST LINE
30 SECOND
72 THIRD
73 FOURTH
150 FIFTH
1000 SIXTH
#
```

```
reseq
#UPDATING
#
```

```
1
100 FIRST LINE
200 SECOND
300 THIRD
400 FOURTH
500 FIFTH
600 SIXTH
#
```

```
reseq 123-456 +5
#UPDATING
#
```

```
1
100 FIRST LINE
200 SECOND
205 THIRD
210 FOURTH
500 FIFTH
600 SIXTH
#
```

# RMERGE

### Syntax

```
— RMERGE —<file title> —<sequence range list> —<sequence range list>—<file title>
```

### Explanation

The RMERGE command copies a file specified as <file title> into the work file, collating the sequence numbers from both files and preserving the <file title> records wherever matching sequence numbers occur.

The <sequence range list> construct limits the RMERGE to specific portions of the <file title>. The default is the entire file.

The UPDATE command is implicitly invoked after the RMERGE command has been executed.

The RMERGE command performs the same function as the MERGE command, except where sequence numbers coincide; in that case, the <file title> version, rather than the work file version, takes precedence.

### Examples

```
list
100 work file - line 1
200 work file - line 2
300 work file - line 3
400 work file - line 4
500 work file - line 5
#
```

```
list rmergefile
#FILE (USERID)RMERGEFILE ON PACKID
100 rmergefile - line 1
150 rmergefile - line 2
200 rmergefile - line 3
250 rmergefile - line 4
#
```

```
rmerge rmergefile
#UPDATING
#
```

```
list
100 rmergefile - line 1
150 rmergefile - line 2
200 rmergefile - line 3
250 rmergefile - line 4
300 work file - line 3
400 work file - line 4
500 work file - line 5
#
```

## RO Command

---

### RO

The RO (reset options) command is described with the SO (set options) command.



## **RUN**

The RUN and EXECUTE commands are synonyms. (Refer to the EXECUTE command.)

# SAME

### Syntax

-- SAME \_\_\_\_\_

### Explanation

The SAME command initiates page mode or refreshes the currently displayed page. If a terminal is not capable of page mode, the LIST command should be used to display one record at a time.

The SAME command also returns CANDE to page mode when page mode sequencing is interrupted by a nonpage-invoking CANDE command. CANDE then displays the page that was displayed just before page mode was interrupted.

# SAVE

## Syntax

```

SAVE [SOURCE | OBJECT | OBJECT AS $ | RECOVERY] [AS <file name>] [NOCRUNCH]

```

## Explanation

The SAVE command saves the current work file and/or its associated object file in the user's library. The work file is saved in the user's library using the current work file name (OBJECT/work file name for the object file). Any other existing file of the same name is removed. If required, the work file is updated before the SAVE command is executed.

After a work file has been saved, it is still available for further editing. The recently saved file serves as the file part of the work file; therefore, it remains present and unmodified until the next SAVE command is executed.

If an attempt is made to save a file that is to overwrite a file whose LOCKEDFILE file attribute is set to TRUE, that file is not saved and the following message is displayed:

```
<file name> NOT SAVED (LOCKEDFILE).
```

For more information about the LOCKEDFILE file attribute, refer to the *File Attributes Reference Manual*. See the ALTER command earlier in this section for information about changing the LOCKEDFILE attribute.

An object file that is saved (but not saved with the AS option) remains associated with the work file and is available to be run or executed until any changes to the work file are entered.

The SAVE AS <file name> form saves the work file in the user's library using the specified file name. If a work object file exists, it is saved as OBJECT/<file name>. A work object file exists if the work file has been compiled but not yet saved. The SAVE SOURCE AS <file name> or SAVE OBJECT AS <file name> forms save only the work source or work object, respectively. The SAVE OBJECT AS \$ <file name> form saves the work object as the specified file name without an OBJECT/ preceding. If file name exists, the SAVE is not performed.

If NOCRUNCH is specified, the files are not crunched when they are saved. The default is to crunch all files.

Entering SAVE RECOVERY saves the work file in the form of a recovery file. Because CANDE does not update in this case, larger files are stored more quickly than with other forms of the SAVE command. The recovery file must be retrieved with the RECOVER command.

## SAVE Command (cont.)

---

### Examples

WHAT  
#WORKFILE A: ALGOL

SAVE  
#UPDATING  
#WORKSOURCE A SAVED

WHAT  
#WORKFILE TEST: ALGOL

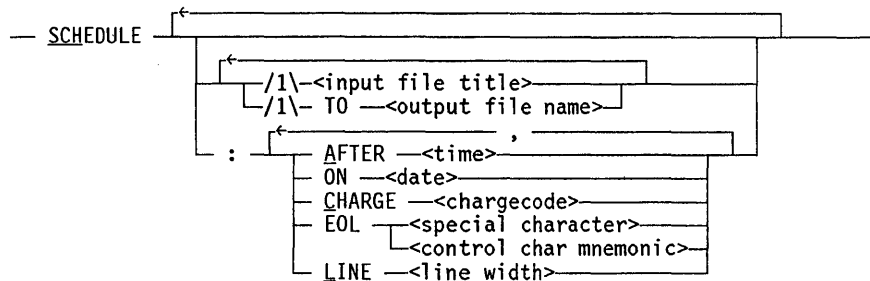
SAVE SOURCE AS NEW/PROG  
#UPDATING  
#WORKSOURCE TEST SAVED AS (UZER)NEW/PROG ON USERPACK

C  
#COMPILING 5084  
#ET=18.8 PT=0.5 IO=0.4

SA OBJECT AS \$TEST  
#WORKOBJECT TEST SAVED AS (UZER)TEST ON USERPACK

# SCHEDULE

## Syntax



<input file title>

—<file title>\_\_\_\_\_

<output file name>

—<file name>\_\_\_\_\_

<date>

—<month> / —<day> / —<year>\_\_\_\_\_

<line width>

—<integer>\_\_\_\_\_

## Explanation

The SCHEDULE command invokes a file of CANDE commands (including program input) in a separate schedule session that is independent of the user's station. The schedule session can begin immediately, or after a specified time and date. The schedule input is merged with any session output to generate a schedule output file, which is an image of the information that would have appeared on a terminal had the session been interactive. The user can inquire about schedule sessions with a STATUS command and can end a session with a STOP command.

<input file title>

The input file title, if specified, indicates the title of the file that contains the schedule-session input; it can specify any file accessible to the user. If no file title is specified, the work file is used. CANDE copies the input into a special schedule file, so the named file or work file can be changed or removed as soon as the SCHEDULE command is completed with no effect on the schedule session. The input file can be of any type (FILEKIND) recognized by CANDE. Only the text portion of each line is processed by CANDE or user programs as input; the text field cannot exceed 84 characters in width.

## SCHEDULE Command (cont.)

---

### <output file name>

The output file name, if specified, indicates the name of the file that is to receive the session terminal image; it cannot have a usercode or asterisk (\*) prefix. If the output file name is omitted, a name is generated by prefacing *SCHOUT/* to the input file title or work file name, respectively. Any usercode prefix or family suffix in the input file title is ignored. The output file name, whether explicit or default, must be unique from any file name already in the user's library. An unnamed work file, such as one accessed from another usercode that has not yet been given a name in the new usercode, can be designated as a schedule file. However, an output file name must be given to the output schedule file; the default file name node *SCHOUT/* is not sufficient in this case. The following error message is given if an output file name is not specified for this case:

```
#UNNAMED WORKFILE REQUIRES AN EXPLICIT SCHEDULE OUTPUT FILE
```

The output file is created with the security set to PRIVATE when the SCHEDULE command is processed. Initially, the output file contains a single line, #SCHEDULE # < schednum > IS SCHEDULED, followed by a blank line. These lines are overwritten when the scheduled session begins. The output file is of type SCHED (FILEKIND = SCHEDULEFILE); because it contains no sequence numbers, CANDE lists it with none. Any other CANDE command (such as GET, FIND, INSERT, WRITE) treats a SCHED file as type CDATA.

### AFTER <time>

By default, a schedule session is immediately considered for processing, subject only to availability of resources. The *AFTER <time>* command option can be used to defer processing until after a specified time of the day. The time is specified as an integer in 24-hour notation; thus, 0000 represents midnight, 0001 represents 12:01 a.m., and 2359 represents 11:59 p.m. Any value less than the current time of day refers to that time the next day.

### ON <date>

The *ON <date>* option can defer the schedule session to a particular date. This option must be used if processing is to be delayed for more than 24 hours. The date format is <month>/<day>/<year>. Each month and day value can be either a 1-digit or 2-digit number, and the year can be either a 2-digit or 4-digit number. For example, the following form of the SCHEDULE command processes a file named SCH/GENLEDGER/OCT, on October 31, 1991 after 6:00 pm:

```
SCHEDULE SCH/GENLEDGER/OCT :AFTER 1800, ON 10/31/91
```

### LINE <line width>

The *LINE <line width>* command option specifies an integer in the range 72 through 255 to set the maximum terminal width (the MAXRECSIZE of the session output file). The default is the input file text length or 72 characters, whichever is larger.

**CHARGE <chargecode>**

The scheduled session runs under the usercode that scheduled the session. The chargecode can be either explicitly set or set by default. The chargecode is the chargecode in effect at the time the SCHEDULE command was processed. The user's default family and language (obtained from the USERDATAFILE) are used to begin the scheduled session; the family and language in effect when the SCHEDULE command was processed are not used. The FAMILY command can be used as part of the schedule input to change the family during the scheduled session. The LANGUAGE command can be used as part of the schedule input to change the language during the scheduled session.

**EOL <special character>**

By default, each input line ends after the last nonblank character. If *EOL <special character>* or *EOL <control char mnemonic>* is specified, the special character or control character can be used to mark the end of an input line by placing it after the last column of valid input on the line. Any further information is ignored because the character effectively ends the line. For example, the specification *EOL \* permits the following input lines:

```
FIX 1230/e./e. \
1460 \
```

The above EOL specification allows a FIX insertion to end in a blank and allows input of a blank line, respectively. To avoid retaining graphic characters on terminals that allow control characters to be entered, choose control characters that do not interfere with the data comm usage of that station. For example, SUB is a suitable control character mnemonic that is entered as control-Z on most ASCII keyboards and can be entered through EBCDIC cards as a 7-8-9 multipunch.

The EOL construct can also be useful for entering comments. For example, the specification *EOL %* would allow the following input line:

```
RUN GENLEDGER/3QTR % THIS PROGRAM PROCESSES THE GENERAL LEDGER FOR
THE THIRD QUARTER
```

**SCHEDULE Input File**

In essence, the schedule input file is a script for a session. The schedule input file must contain every input line that the user would type at the terminal (after logging on) if the session were interactive. The schedule input file can contain any CANDE command except HELLO, PASSWORD, or TAPE. Any TERM command must be empty (containing no change to terminal specifications). A BYE command is optional and terminates the session wherever it appears. CANDE infers a ?END or BYE, if appropriate, at the end of the schedule input file. A line beginning with a question mark (?) is treated as a control command; the control commands acceptable in a schedule session are defined in subsequent text in this section.

A line entered in single-line sequencing mode that begins with two question marks (??) is treated as valid data beginning with a question mark (?), as in interactive CANDE

sessions. Also, a program reading a line beginning with two question marks (??) discards the first question mark (?); the line is then given as data to the program. This feature is not available to an interactive session.

### REMOTE Files

A schedule session is run with a dummy data comm station to provide for REMOTE files. For most purposes, the schedule station behaves programmatically like a real station. Specifically, the REMOTE files of any tasks processed from the schedule session are associated by default with the schedule station. As with any session, this linkage is accomplished through the STATION task attribute and may be overridden by explicit user action. By setting STATION to a valid logical station number (LSN) or by setting it to zero and equating the file TITLE appropriately, a task of a schedule session may attempt to open a file on a genuine data comm station (subject to the same constraints as any other *foreign* user of a station). On the other hand, only tasks of the particular schedule session can open REMOTE files to a schedule station.

### Schedule Control Commands

The following control commands are available for schedule as well as interactive sessions:

Control Command	Effect
?DENY	Closes all input and output remote files for a program.
?END	Denotes the end of input for a program.
?TIME	Shows current time and date.
?%	Enters comments.

The following special control commands are available only for schedule sessions:

Control Command	Effect
?REPORT	Sends a message to an interactive user.
?NUMBERED	Lists sequence numbers of the input file in the output file.
?UNNUMBERED	Does not list sequence numbers of the input file in the output file.
?RESTART	Restarts aborted sessions.
?NORESTART	Does not restart aborted sessions.
?RESUME	Resumes input after schedule session errors are detected.
?EOL	Changes the end-of-line character.

For a description of these commands, refer to Section 4, "CANDE Control Commands." No control commands other than those listed above are permitted in a schedule input file.



Each schedule session is assigned a unique integer for identification and operator interaction, called *schednum*. Schednum is always displayed as a 5-digit number (with leading zeros) to distinguish it from a session number. The schednum is assigned when the schedule input file is created and is included in the schedule input file title. The schednum appears in the output file and in messages about the schedule session. A session number is assigned when the schedule session processing begins; if the session is restarted or SPLIT, a new session number is assigned. All session numbers used by a schedule session appear in the second line of the output file, as well as in the separate log-on or session-split messages.

### Examples

```
SCH SCH/WFL
#SCHEDULE OUTPUT FILE NAME IS IN USE
#FILE:(UZER)SCHOUT/SCH/WFL ON USERPACK
#
```

```
REMOVE SCHOUT/SCH/WFL
#(UZER)SCHOUT/SCH/WFL ON USERPACK REMOVED
```

```
SCH SCH/WFL
#SCHEDULE #00009 RUNNING
  OUTPUT FILE IS (UZER)SCHOUT/SCH/WFL ON USERPACK
  LINE WIDTH IS 72
#
```

```
REMOVE SCHOUT/SCH/WFL
#(UZER)SCHOUT/SCH/WFL ON USERPACK REMOVED
```

```
SCH SCH/WFL :AFTER 1430, ON 10/31/91
#SCHEDULE #00010 IS SCHEDULED AFTER 14:30 ON 10/31/91
  OUTPUT FILE IS (UZER)SCHOUT/SCH/WFL ON USERPACK
  LINewidth IS 72
#
```

**SCHEDULE Command (cont.)**

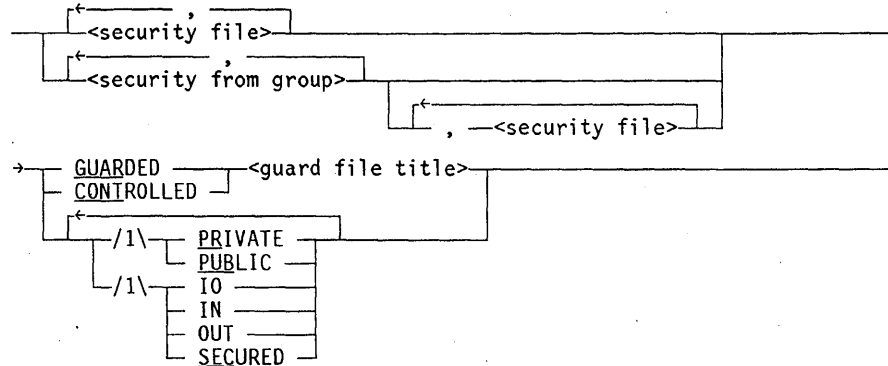
---

# SECURITY

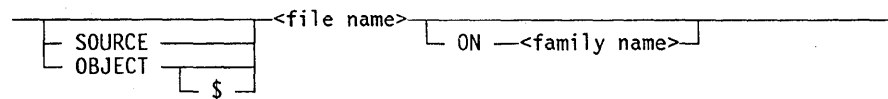
## Syntax

— SECURITY —<security specification>

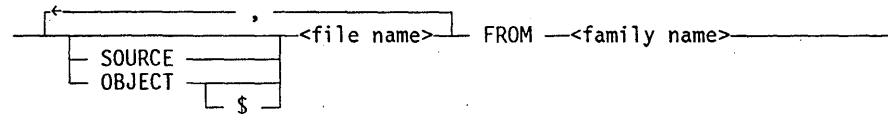
### <security specification>



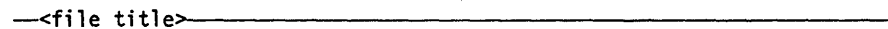
### <security file>



### <security from group>



### <guard file title>



## Explanation

The SECURITY command changes the security attributes of files on disk. If one of the files specified is the work file, the security of the work file will also be changed.

If a <family name> is specified, the command applies only to the <file name> on the volume associated with the family name. For example, SECURITY FILEX ON DISK PUBLIC affects only files found on DISK even though the family specification for the session is FAMILY DISK = USERPACK OTHERWISE OTHERPACK.

If a <family name> is not specified, the command applies to the file name found on DISK. If DISK is also the <target family> in the family specification for the session,

## SECURITY Command (cont.)

---

then the only files affected by the command are those on the < substitute family >. Files on the < alternate family > are not affected.

If the < guard file title > specifies a family name, CANDE searches for the guard file only on the specified family. For example, a < guard file title > of (BILLIE)ENGUARD ON DISK is searched for only on DISK, even if the session has a family specification of FAMILY DISK = BUOYS OTHERWISE GULLS.

If the < guard file title > does not specify a family name, CANDE searches for the guard file on the family DISK. If DISK is also the target family in the family specification for the session, CANDE searches for the guard file on the substitute family, and if necessary, also on the alternate family.

When a family name is specified in the SECURITY command, it is used even if there is a family specification for the session. If no family name is specified and there is a family specification for the session, with DISK as the target family, then a file that is to be altered (the target of the command) is affected only on the substitute family. A guard file, which is not to be altered, is searched for on both the substitute family and the alternate family.

If the guard file is found, its full title, including the usercode or asterisk (\*) prefix, and family name is stored as the value of the target file's SECURITYGUARD attribute. If the guard file is in fact not a guard file, an error message is displayed.

If the guard file is not found, the usercode and primary family of the user who entered the SECURITY command are added to the guard file name, unless the user specifies otherwise, and the resulting title is stored as the value of the target file's SECURITYGUARD attribute.

In a < security from group > construct, a FROM < family name > clause applies to all the file names in that construct. An ON < family name > clause applies only to the immediately preceding file name. An ON clause cannot precede a FROM clause in the same command.

When specifying a file name, a user logged on under a nonprivileged usercode cannot enter a usercode specification other than the usercode for the current session.

If a < guard file title > is specified for the work file, its name must be less than 137 characters, including a prefix of either a usercode or an asterisk (\*), and an ON < family name > suffix. A prefix or suffix is added by CANDE if not supplied in the command.

CLASSA is allowed as a nonpreferred synonym for PUBLIC.

If SOURCE or OBJECT is used as a file name, it must be enclosed in quotation marks to distinguish it from the keywords SOURCE and OBJECT.

The SECURITY < file name > form specifies the file for which the security attributes are to be changed. The security attributes are changed for the file with the specified file name and any associated object file.

The SECURITY SOURCE < file name > form changes the security attributes of the file with the specified file name but does not affect the object file associated with the file name.

The SECURITY OBJECT < file name > form changes only the attributes of the object file associated with the file name. If a dollar sign (\$) follows OBJECT, it indicates that the object file whose security is to be changed is stored as file name, not as OBJECT/< file name >, so only the file with the specified file name has its security changed.

The security options that can be specified are identical to the values of the file attributes SECURITYUSE, SECURITYTYPE, and SECURITYGUARD. The values for SECURITYTYPE are PRIVATE, PUBLIC, GUARDED, and CONTROLLED. The values for SECURITYUSE are SECURED, IN, OUT, and IO. SECURITYGUARD identifies the guard file title. For additional information, refer to the SECURITYUSE, SECURITYTYPE, and SECURITYGUARD attributes in the *File Attributes Reference Manual*.

The default security values for all CANDE-created files are PRIVATE and IO.

### Examples

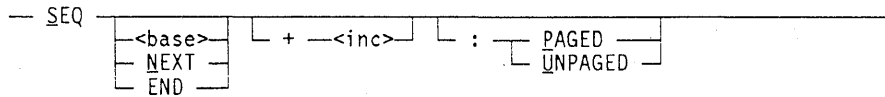
```
SECURITY OBJECT MY/EDITOR, TEST PUBLIC IO
#(UZER)OBJECT/MY/EDITOR ON USERPACK SECURITY CHANGED
#(UZER)TEST ON USERPACK (& OBJECT) SECURITY CHANGED
```

```
SECURITY F,G FROM OTHERPACK PRIVATE
#(UZER)F ON OTHERPACK SECURITY CHANGED
#(UZER)G ON OTHERPACK SECURITY CHANGED
```

```
WHAT
#WORKFILE A: ALGOL, 10 RECORDS, SAVED
SECURITY A,B,C PUBLIC
# (JOE)A ON UPACK (& WORKFILE) SECURITY CHANGED
# (JOE)B ON UPACK SECURITY CHANGED
```

## SEQ

### Syntax



### Explanation

The SEQ command causes the system to automatically provide a sequence number for each line to be entered in the work file. Sequencing is done either in single-line sequencing mode (UNPAGED) with one line number presented at a time, or in page-mode sequencing (PAGED) with a whole page of numbered lines presented. The default setting for SEQ on screen terminals capable of page mode is PAGED, while nonscreen terminals have only single-line sequencing (UNPAGED) available.

The SEQ command can be used either when a file is created or while editing a pre-existing file. When a file is created, SEQ provides sequence-numbered lines with a specified base and increment on which to enter text. While editing a pre-existing file, SEQ can be used to provide additional numbered lines at the end of the work file (SEQ with the UNPAGED option, SEQ NEXT with the PAGED option), or it can be used to insert lines at a newly specified base and increment, within the preexisting text (see the second example under PAGED at the end of this command description).

The following paragraphs, with the exception of those captioned PAGED and UNPAGED, apply to both single-line sequencing mode and page mode sequencing.

The starting line number can be specified in one of four ways, as defined in the following paragraphs:

The SEQ command without <base>, NEXT, or END gives a beginning line number of 100 when beginning a new file. When adding to an existing file, SEQ without <base>, NEXT, or END always displays line 100 for the PAGED option. For the UNPAGED option, SEQ used without <base>, NEXT, or END works the same as SEQ NEXT (described further on).

#### <base>

The <base> construct explicitly defines the starting line number.

#### NEXT

The NEXT option sets the starting line number to the next line number that would have resulted from the last SEQ, RESEQ, MOVE, or INSERT command. If none of these commands have been issued, then 100 is used.

**END**

The END option sets the starting line number to the largest line number in the work file plus the specified (or default) increment. If no specifications appear, 100 is used as the increment.

**<inc>**

The <inc> value specifies an increment to be used in generating subsequent line numbers. 100 is the default unless an increment has been specified in a prior SEQ, RESEQ, MOVE, or INSERT command.

**PAGED**

The PAGED option applies only to SCREEN terminals with the minimum of PAGESIZE and MAXOUTPUT/LINEWIDTH greater than two lines. The MAXINPUT must be greater than or equal to the smaller of MAXOUTPUT, and PAGESIZE\*LINEWIDTH. PAGED is the default for terminals with these characteristics. If PAGED is used on a terminal that does not have these characteristics, an error message is given. The TERMINAL command in this manual lists the MAXOUTPUT, LINEWIDTH, and MAXINPUT.

The PAGED option sends a page with sequence numbers at the left of the screen to the user. NEXT and a column indicator appear at the top of the screen and the user can enter and transmit data back to CANDE one page at a time. If no command other than VOID is transmitted within the page, CANDE sends a new page of sequence numbers that begins with the last record from the previous page.

The SEQ command with the PAGED option puts page-mode-capable terminals into page mode sequencing. CANDE sends the user a page with sequence numbers at the left of the screen, as described above. The shortened versions of the MARGIN and FIX commands, which can be inserted within a line with single-line sequencing, cannot be used with page mode. Any command, with the exception of VOID, transmitted after a SEQ command automatically stops page mode sequencing. This is unlike single-line sequencing mode (the UNPAGED option) where the only way to send a command is to first break the sequence by sending a null input. See also UNPAGED in this section for more information.

If one or more commands are transmitted within the page, sequencing is discontinued and the commands are executed. The SEQ NEXT form resumes sequencing where it left off. Transmitting SEQ alone gives a numbered page of text beginning with line number 100. This result is different from the result of transmitting SEQ alone using the UNPAGED option. For more information about page mode commands, refer to "Page Mode Operations" in Section 1, "General Information."

### UNPAGED

The UNPAGED option is the default for non-SCREEN type terminals (a type declared in NDLII that can be modified by the TERMINAL SCREEN/HARDCOPY command) and for SCREEN type terminals that do not have the capability of page mode. SCREEN type, page-mode-capable terminals can use this option to initiate single-line sequencing mode.

If the UNPAGED option is specified, single-line sequencing is invoked and the system recognizes only the commands that begin with the defined control character of the station (control commands) and a special adaptation of the MARGIN and FIX commands; all other information is treated as text. Single-line sequencing continues until you press the ETX key followed by the XMIT key. (On non-TD terminals, you press the RETURN key.) Pressing the keys sends a null line. Refer to the MARGIN and FIX commands in this section for a complete description of these commands.

In single-line sequencing mode on a teletype or similar device, the new line number is typed at the beginning of each line; the text desired for that line is then entered. On devices with multiline input capability, the lines must be separated with carriage returns. A new sequence number is displayed after each transmission; the line number is incremented but not displayed for subsequent lines within a transmission block.

Normal CANDE commands are not recognized in single-line sequencing mode because they are indistinguishable from text entries. If the line begins with the defined control character, usually a question mark (?), the input is processed as a control command. If the defined control character is to be entered as the first character of a line in sequence mode, the line must be preceded by an extra control character. That is, the control character must appear twice (??) if it is to be entered as the first character. This also applies to the asterisk (\*) and at sign (@).

Although sequence numbers are generated, the line is otherwise treated as a single-line entry. That is, if the new line number matches one in the work file, the newer line replaces the older. This can result in accidentally overwriting existing lines. The SEQ END command always adds new lines to the end, but the SEQ NEXT command might overwrite lines.

Different types of terminals may act differently in single-line sequencing mode; therefore, the user should learn the characteristics of the particular terminal to be used.



**Examples****PAGED**

In the following example, page mode sequencing using the PAGED option is displayed. The following page of line numbers is presented to the user simultaneously. (This is a sample with an ALGOL file.)

```

SEQ 20 + 10
NEXT+ ....*....1....*....2.. ... ..*....6....*....7..
00000020
00000030
00000040
00000050
.
.
.
00000250

```

Occasionally, the user may want to insert blank lines into a set of records. In the following example, records with sequence numbers 100, 110, 120, 200 and 300 already exist.

```

SEQ 50 + 50
NEXT+ ....*....1....*....2.. ... ..*....6....*....7..
00000050
00000100ABC
00000110DEF
00000120GHI
00000150
00000200JKL
00000250
00000300MNO
.
.
.
00001150

```

## SEQ Command (cont.)

---

### UNPAGED

In the following examples, single-line sequencing mode using the UNPAGED option displays the following lines one at a time:

```
S
100YOUR STATEMENTS
200@5
300TEST COLUMNS
400
#
```

```
L
100 YOUR STATEMENTS
200    TEST COLUMN
#
```

```
seq 10+3
10 your statements....
13 and so on....
16 and so on....
19
#
```



## Single-Line Entry or Deletion (cont.)

---

If the current margin setting is 10, then text begins in column 15.

**Note:** *To permanently set the margin to a particular column, the MARGIN command must be used.*

### Examples

```
MAKE TEST1 DATA
#WORKFILE TEST1:SEQ
25begin
100this is a single-line entry
215and so is this
25begin
30005:this text starts in column 5
3150+10:this text starts in col. 11 (col. 1 + 10 = 11)
3170@ this text begins with a single "@" character
000003208 digits maximum in type seqdata files
```

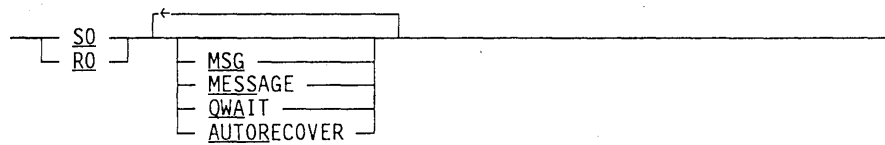
```
list
#WORKFILE TEST1
25 begin
100 this is a single-line entry
215 and so is this
300   this text starts in column 5
315       this text starts in column 11 (col 1 + 10 = 11)
317 @ this text begins with a single "@" character
320 8 digits maximum in type seqdata files
#

215
300line 215 was just deleted, and this replaces the line at 300
```

```
list
#WORKFILE TEST1
25 begin
100 this is a single-line entry
300 line 215 was just deleted; and this replaces the line at 300
315       this text starts in column 11 (col 1 + 10 = 11)
317 @ this text begins with a single "@" character
320 8 digits maximum in type seqdata files
#
```

## SO or RO

### Syntax



### Explanation

The SO and RO commands control the setting and resetting of session options. SO sets session options, and RO resets session options. The current state of session options is displayed if either the RO or SO command is entered without a following session option name.

MESSAGE and MSG are synonyms. These options control the automatic display of job-related and task-related messages at the terminal during a CANDE session. If MSG is set, job-related and task-related messages are automatically displayed. When MSG is reset, only RSVP and DS messages for a task running at the user's station are displayed. The default value of the MSG option is determined by the CANDEGETMSG Boolean setting for the usercode in the SYSTEM/USERDATAFILE.

The QWAIT option specifies that the state of the user's queued input is to be set to waiting rather than pending in the event of an error. (Refer to "Input Queue Manipulation" in Section 1, "General Information," for details.) The default value of the QWAIT option is determined by the CANDEQWAIT Boolean setting for the usercode in the SYSTEM/USERDATAFILE.

The AUTORECOVER option controls the action of CANDE when processing any recovery file, including the recovery file discovered after successfully logging on. The default value for the AUTORECOVER option is determined by the CANDEAUTORECOVER Boolean setting for the usercode in the SYSTEM/USERDATAFILE and the value of the CANDE configuration option RECOVERSESSION. If the option CANDEAUTORECOVER is not initialized for the specified usercode, then the default value used is RESET. The default value of AUTORECOVER for a session can be determined by the following table:

RECOVERSESSION Option	CANDEAUTORECOVER SET	CANDEAUTORECOVER RESET
ALL	TRUE	TRUE
NONE	FALSE	FALSE
REQUESTED	TRUE	FALSE

## SO Command (cont.)

---

The SO and RO commands are also available as control commands (?SO and ?RO). The only difference is that the control forms can be entered when a remote file is open or the station is busy, whereas the noncontrol forms are queued if entered while the station is busy.

### Examples

```
SO
#MESSAGES SET, QWAIT RESET, AUTORECOVER RESET
```

```
so messages
#MESSAGES SET
```

```
SO QWAIT
#QWAIT SET
```

```
SO AUTORECOVER
#AUTORECOVER SET
```

```
RO
#MESSAGES SET, QWAIT SET, AUTORECOVER SET
```

```
ro messages
#MESSAGES RESET
```

```
RO QWAIT
#QWAIT RESET
```

```
RO AUTORECOVER
#AUTORECOVER RESET
```

```
RO
#MESSAGES RESET, QWAIT RESET, AUTORECOVER RESET
```

# SPLIT

## Syntax

— SPLIT —————|

## Explanation

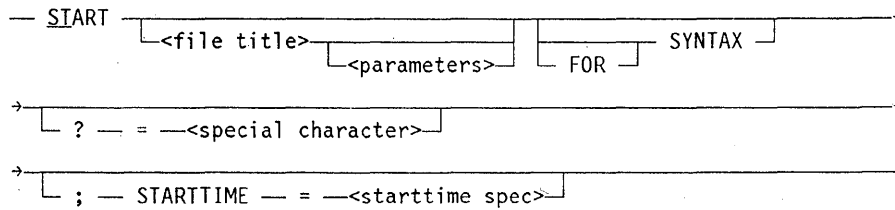
The SPLIT command terminates the current session and automatically begins another session under the same usercode. If the current session has produced any printer or card punch output, that output is queued for printing or punching at this time. The work file, if any, is not disturbed.

## Example

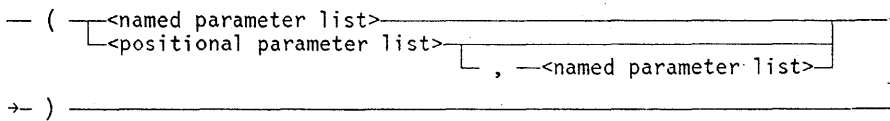
```
SPLIT  
#SPLIT SESSION 7562 ET=1:23:57.3 PT=13.6 IO=6.8  
#NEW SESSION 7695 11:08:20 05/08/78
```

# START

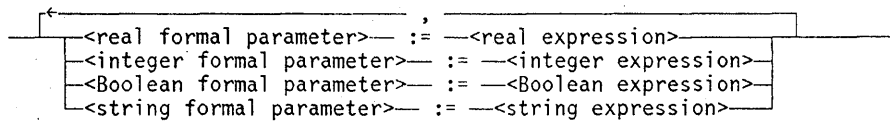
## Syntax



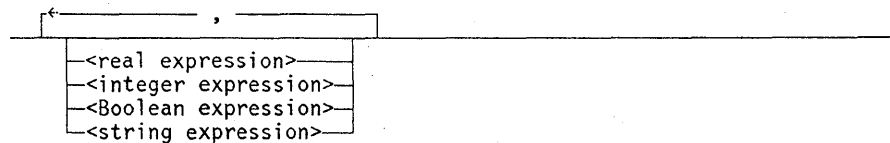
### <parameters>



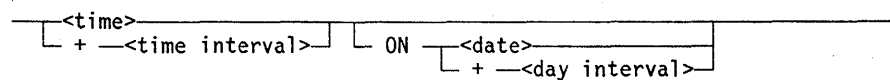
### <named parameter list>



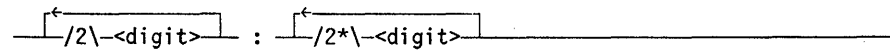
### <positional parameter list>



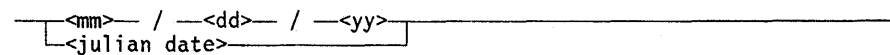
### <starttime spec>



### <time>



### <date>

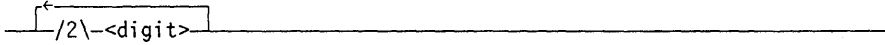


### <julian date>

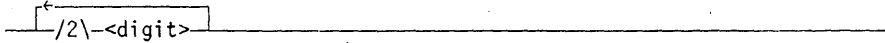




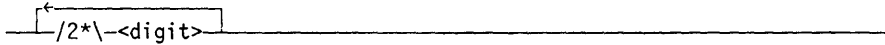
<mm>



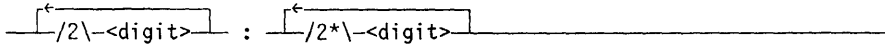
<dd>



<yy>



<time interval>



<day interval>



**Explanation**

The START command instructs the WFL compiler to process the specified < file title > . If the START command is specified without a < file title > , the current work file is processed. On the Mark 3.8 system software release and later releases, the START command must not be followed by another command.

The < parameters > construct is any arbitrary list of constants (that is, integer, real, string, or Boolean) recognized by WFL. It can be included in the START command if the job being initiated includes a job parameter list.

Parameters can be passed by listing them in positional order (positional parameters) or by explicitly naming them (named parameters). Named parameters can be listed in any order, and are passed by the corresponding formal parameter in the job parameter list. An example is START JOB/1(JOBQUEUE := 4).

An actual parameter is not required to be passed if the corresponding formal parameter specifies the keyword OPTIONAL. If an actual parameter is not passed, the default value is assigned as follows:

Type	Default Value
Boolean	FALSE
Integer	0
Real	0
String	""

A default value can also be assigned by using the DEFAULT clause for the corresponding formal parameter in the WFL job.

When using parameters with default values, an item such as a default queue or a default chargecode can be specified for a job and passed only when another queue or chargecode is desired.

When using optional parameters, default values can be specified for optional job parameters if values other than the standard defaults are desired.

If a <positional parameter list> is used, optional parameters can be omitted by specifying consecutive commas or by specifying fewer actual parameters than formal parameters. When fewer parameters are passed than expected by the job, the remaining parameters in the job parameter list must be specified as optional. For additional information about the <positional parameter list>, refer to the *WFL Reference Manual*.

Combinations of parameter usage can be applied. Named parameters can be used with positional parameters. Positional parameters are listed in their normal position, followed by the named parameters. Once a named parameter is used, the remaining parameters must also be named parameters. This combination is particularly useful when the required job parameters are declared before optional parameters. An example is

```
START JOB/2("OBJECT/X", JOBQUEUE: =4).
```

Named parameters can be used with optional parameters. For example, if a job has many parameters, the parameters can be specified as optional. The job can then be started by explicitly naming the parameters in the start parameter list. In such cases, only the necessary parameters are passed. For additional information about the <named parameter list>, refer to the *WFL Reference Manual*.

The SYNTAX option specifies that the job is to be compiled only for syntax checking; the job will not be executed. When a job is compiled for syntax, it is not necessary to pass parameters to the job, even though the job is specified with parameters. However, WFL does check for the accuracy of the types of parameters if they are supplied. By default, the question mark (?) is interpreted by WFL to be an invalid character when found in column 1 of an input line. This default can be changed by specifying ? = <special character>, where <special character> represents the character that WFL is to interpret as the invalid character. (The invalid character is the equivalent of a 1-2-3 multipunch in an EBCDIC card deck.)

The resulting job is invoked separately from the session, although progress can be monitored through use of commands such as ?JA, ?MIX, ?C, and ?MSG.

The invoked WFL compiler determines the absolute time and date at which the job should begin execution from the <starttime spec> construct. This start time overrides any start time specified in the <job attribute specification> list of the job that is being started. If the <starttime spec> is not supplied, the job is accepted for immediate processing. Refer to the *WFL Reference Manual* for information about job attribute specifications.

The <time> construct is the time of day on a 24-hour clock in the form HH:MM. HH, the hour, must be less than 24, and MM, the minute, must be less than 60.

The < date > construct is the date in the form MM/DD/YY or the julian date. The julian date is the date in the form YYDDD, where YY is the year and DDD is the number of the day of the year.

The < time interval > construct is of the form HH:MM. HH, the number of hours, must be less than 24, and MM, the number of minutes, must be less than 60.

The < day interval > construct specifies a number of days to be added to the current date.

A STARTTIME specification is accepted, but ignored, if the started job is a WFL job created before the Mark 2.9 system software release.

### Examples

```
START WFL/1
#RUNNING 7768
#JOB 7769 IN Q 04
#
#7769 BOJ TESTJOB
#7769 DISPLAY:IN START JOB
#7769\7769 EOJ JOB TESTJOB
```

```
L TEST
#FILE (UZER)TEST ON USERPACK
100 BEGIN JOB TEST/PARAM (INTEGER I, STRING S);
200 INTEGER J;
300 DO
400 BEGIN
410 DISPLAY S;
420 J:=J+1;
430 END
500 UNTIL J=I;
600 END JOB
#
```

```
START TEST(2,"HELLO")
#RUNNING 4833
#JOB 4834 IN Q 04
#
#4834 BOJ TEST/PARAM
#4834 DISPLAY:HELLO.
#4834 DISPLAY:HELLO.
#4384
#4384\4384 EOJ TEST/PARAM
```

## START Command (cont.)

---

```
START WFL/EX/1; STARTTIME= 23:00
#RUNNING 1101
#JOB 1102 IN QUEUE 0
#
```

```
L WFL/PARAMS
#FILE (JOE)WFL/PARAMS ON USERS
100 BEGIN JOB WFL/PARAMS(String CODEFILE,
200             INTEGER JOBQUEUE OPTIONAL DEFAULT=3,
300             BOOLEAN BIND OPTIONAL);
350     CLASS = JOBQUEUE;
400 DISPLAY "CODEFILE = " & CODEFILE;
500 DISPLAY "JOBQUEUE = " & STRING(JOBQUEUE, *);
600 IF BIND THEN
700     DISPLAY "BINDING = TRUE"
800 ELSE
900     DISPLAY "BINDING = FALSE"
950 END JOB
#
```

```
START WFL/PARAMS("OBJECT/A")
#RUNNING 9023
#JOB 9023 IN QUEUE 3
#9023 BOJ EXAMPLE
#9023 DISPLAY:"CODEFILE = OBJECT/A".
#9023 DISPLAY:"JOBQUEUE = 3".
#9023 DISPLAY:"BINDING = FALSE".
#9023 EOJ EXAMPLE
```

```
START WFL/PARAMS(JOBQUEUE := 5, CODEFILE := "OBJECT/B")
#RUNNING 9043
#JOB 9043 IN QUEUE 5
#9043 BOJ EXAMPLE
#9043 DISPLAY:"CODEFILE = OBJECT/B".
#9043 DISPLAY:"JOBQUEUE = 5".
#9043 DISPLAY:"BINDING = FALSE".
#9043 EOJ EXAMPLE
```

```
START WFL/EX/2; STARTTIME= + 0:05
#RUNNING 1103
#JOB 1104 IN QUEUE 0
```

```
START WFL/EX/3; STARTTIME = 1:30 ON 08/29/81
#RUNNING 105
#JOB 1106 IN QUEUE 0
#
```

```
START WFL/EX/4; STARTTIME = 1:00 ON + 2  
#RUNNING 1107  
#JOB 1108 IN QUEUE 0  
#
```

# STATUS

### Syntax

— STATUS —<output file name>—————|

### Explanation

The STATUS command can be used to inquire about any schedule session initiated by the user. If the session is not yet in progress, “#SCHEDULE # <schednum> IS SCHEDULED” is displayed; if AFTER <time> is pending, that information is also displayed.

If the session is in progress, information similar to the following is returned:

```
#SCHEDULE #00017 STARTED AT 12:34
READ 12 (THRU 2100), WRITTEN 146, RESUMED AFTER 2 ERRORS
SESSION 5214, TASK/<LSN> 5215/119
```

The < schednum >, start time, lines read and written, sequence number last read, and session number always appear. The error count appears if it is nonzero.

If the session has been completed, a message indicates normal termination. If the session has been abnormally terminated, a message indicates the nature of the abnormal termination.

A STATUS inquiry made during the first minute after CANDE is restarted may show a status like “#SCHEDULE #00008 STARTED AT 09:27”. No further information is displayed because the session was running when CANDE was aborted and the schedule facility has not yet been reinitialized. As soon as schedule initialization is completed, the status is changed to “AWAITING RESTART” or “ABORTED BY SYSTEM FAILURE; NO RESTART”.

**Note:** *The minimum abbreviation is STAT. Further abbreviation produces a START command. The STATUS command is distinct from the ?STATUS control command.*

### Example

```
STATUS SCHOUT/SCH/10
#SCHEDULE #00014 STARTED AT 13:07
READ 5 (THRU 500), WRITTEN 19
SESSION 7958, TASK/<LSN> 7960/39
```

# STOP

## Syntax

```
-- STOP --<output file name>-----|
```

## Explanation

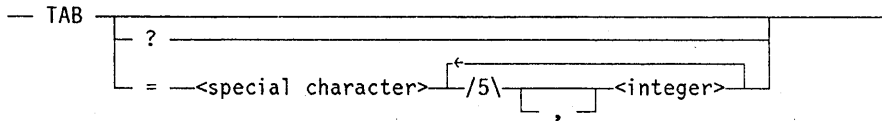
The STOP command causes the schedule session associated with <output file name> to be aborted or removed from the queue if it is in progress or pending, respectively.

## Example

```
STOP SCHOUT/SCH/10  
#SCHEDULE #00014 BEING STOPPED
```

## TAB

### Syntax



### Explanation

The TAB command allows the user to define a tab character and set tab columns. These tabs have no relation to the TAB functions on a device. The <special character> can be any special character except the question mark (?), the at sign (@), and the asterisk (\*).

The <integer> values specify the columns in which the cursor is to be placed (that is, the same numbers as in the MARGIN command). The user can insert tab characters anywhere in the source line to advance to the next tab column. To advance to the first tab column, a single tab character is inserted in the text. To advance to the second column, two tab characters are inserted unless a tab character had been inserted previously to advance to the first tab column. In the latter case, only one tab character is inserted. If the number of tab characters recognized indicates a backwards skip, a warning message is displayed and the tab is ignored. If CANDE is in sequence mode when a backwards skip takes place, sequence mode is terminated (as in a line overflow).

The TAB command clears the current tab settings. The TAB? form displays the current tab settings. If the character is not a graphic character, its name or HEX representation is displayed (for example, HT, VT, CB).

Tabs are cleared at each new log on or when the TAB command is entered by itself.

### Examples

The number sign (#) indicates a blank character; input is to a work file.

```
COMMAND: TAB = ' 5,10,15 [TABS CHAR=', TABS AT COL 5,10,15]
```

1. INPUT: 'ABC  
ACTION: ####ABC
2. INPUT: 'ABC'DEF  
ACTION: ####ABC##DEF
3. INPUT: ABCD'EFG  
ACTION: ABCDEFG
4. INPUT: ABCDE'FG  
ACTION: ABCDEFG  
WARNING: #TAB COL(5) < CURRENT COL(6) - IGNORED.



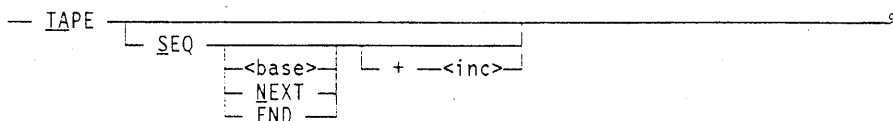
5. INPUT: ABCDE''FG  
ACTION: ABCDE####FG

COMMAND: TAB?  
RESPONSE: #TAB = ' 5,10,15

COMMAND: TAB  
RESPONSE: #TABS CLEARED

# TAPE

## Syntax



## Explanation

The TAPE command initiates the reading of data from paper tape. The data is entered in the work file as the paper tape is read.

After the TAPE command is entered, CANDE sends "#OK" followed by a carriage return (CR), line feed, and DC1 (XON). The XON automatically starts the paper tape reader on some terminals; on other terminals, the reader must be manually started within 15 seconds. If 15 seconds elapse with no input received, the data comm subsystem signals an error termination and CANDE sends a message to the terminal. Once input has begun, it continues until the paper tape reader stops (as noted by a 500-millisecond period with no new character received). During input, a CR delimits a work file line. The following character codes are ignored or discarded:

Character Code	Function
LF	Line Feed
NUL	Null
DEL	Rubout
DC1	XON
DC3	XOFF

Sequence numbers are optional on the input tape; however, if the tape does not contain sequence numbers, the SEQ option must be used. The SEQ option causes CANDE to supply sequence numbers for all lines. (Refer to the SEQ command in this section for explanations of <base>; <inc>, NEXT, and END.)

Because the text field begins at the first nondigit or the first character after the maximum number of digits for a sequence number, leading zeros in sequence numbers must be supplied only if the first character of text is a digit.

## Input Errors

Paper tape input is terminated if no input appears for 15 seconds. The input following the error is discarded under any of the following conditions:

- A break is detected on the line.
- A record exceeds maximum input length.
- The data communications subsystem fails to acquire message space when needed.

A stop bit error is treated by substituting a question mark (?) for the garbled character; input is not aborted.

### Offline Preparation of Paper Tape Input

The minimum requirement for CANDE input is that each line be terminated by a CR. In order for the output to be readable on the teletypewriter as the tape is being read, each line should be terminated by CR, LF, DEL (rubout), or NUL. The LF advances the paper, and the rubout or NUL gives the carriage time to return. Leader or trailer tape can be punched using the repeat key with DEL or NUL. (NUL can be punched on many teletypewriters by using shift, control, and P.) Errors committed in punching the tape can be corrected by backspacing the paper tape, typing rubout, and then typing the correct character. (Usually a backspace exists on the paper tape punch for this purpose.)

To have the system punch a paper tape of an existing file, use the PUNCH option of the LIST command.

*Note: Special NDII code is required in order for this command to be correctly processed on the current equipment.*

### Examples

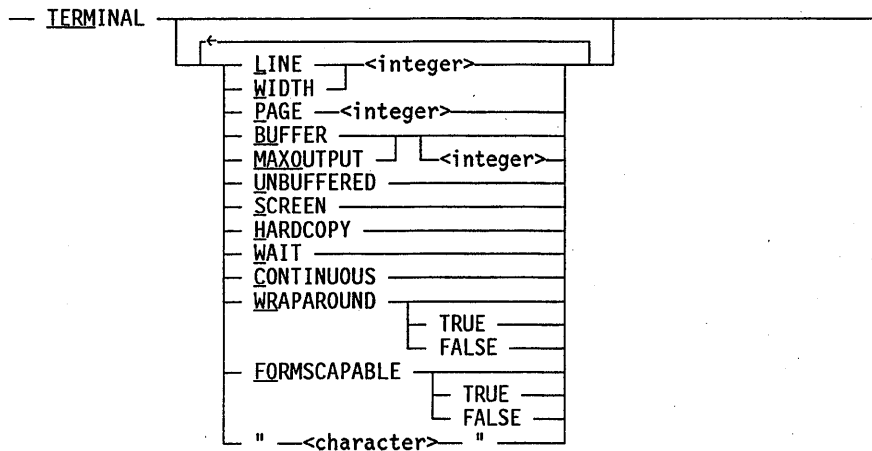
```
M DON/1
#WORKFILE DON/1: SEQ
TAPE SEQ
#OK
LIST
100 THIS WAS AN EXAMPLE.
200 END.
#
```

```
MAKE T; TAPE
#WORKFILE T: SEQ
#OK
#WORKFILE T
00000100THIS IS THE FILE BEING READ IN
00000200IT WAS LISTED WITH THE L : PUNCH COMMAND
```

```
#1 RECORDS DISCARDED (NO SEQUENCE NUMBER)
#
L
100 THIS IS THE FILE BEING READ IN
200 IT WAS LISTED WITH THE L : PUNCH COMMAND
#
```

# TERMINAL

## Syntax



## Explanation

The TERMINAL command specifies attributes of the terminal, which determine the CANDE data transmission mode. The specifications are significant during the execution of CANDE output commands such as LIST and FIND, and for determining whether a terminal is page-mode capable.

The WIDTH option specifies the width of the terminal line in characters. WIDTH and LINE are synonyms.

The PAGE option specifies the number of lines per page or screen.

The BUFFER option indicates that the terminal is buffered and, optionally, specifies the number of characters the terminal can receive and display. If < integer > is absent, the previous default specification is used.

MAXOUTPUT is a synonym for BUFFER. Refer to "Page Mode Operations" in Section 1, "General Information" for its application.

The UNBUFFERED option indicates that the terminal is unbuffered. The buffer size is not lost but is ignored.

The SCREEN option indicates that the terminal is a screen device.

The HARDCOPY option indicates that the terminal produces permanent output, such as a teletype.

The WAIT option causes CANDE to wait after sending each page until a signal is sent to display the next page. The signal is a null (empty) line.

The CONTINUOUS option causes CANDE to send continuously until end-of-output.

When an output line exceeds the terminal width, CANDE either truncates the line or "folds" it by printing it on several lines. A special foldmark character is printed as the last character of a truncated or interrupted line and as the first character of a continuation line. The foldmark character can be specified by placing it within quotation marks in the TERMINAL specification command. If the quotation marks enclose more than one character, the first is used.

The WRAPAROUND option indicates whether or not the terminal performs an automatic line feed and carriage return when a character is displayed in the last character position of a line. If WRAPAROUND is TRUE, page mode output the size of a terminal line is sent without a carriage return and linefeed. If WRAPAROUND is FALSE, page mode output that is less than or equal to the size of a terminal line is sent with a trailing carriage return and linefeed. Trailing blanks are suppressed. WRAPAROUND is set by NDLII and only in rare cases needs adjustment using the TERMINAL command.

If the TERMINAL command is entered with no options specified, CANDE displays the current terminal settings. When modifications are made with the TERMINAL command, CANDE displays the terminal settings including any modifications.

The FORMSCAPABLE option is used to identify the terminal as being capable of handling TD, MT, and ET control codes for cursor placement and forms editing. If FORMSCAPABLE is set to TRUE, the cursor is positioned after the NEXT +/- prompt in page mode. The initial setting of FORMSCAPABLE is FALSE for all terminals, but once the terminal setting is changed, CANDE retains that terminal setting for that CANDE session.

The MAXINPUT option specifies the number of characters the terminal can transmit. There is no way to modify this value through CANDE. The fixed value for MAXINPUT is displayed as one of the current terminal settings. Refer to "Page Mode Operations" in Section 1, "General Information" for further information.

### Examples

```
TERMINAL
#LINE = 80, PAGE = 24, MAXOUTPUT = 1920, MAXINPUT = 1920, SCREEN,
WAIT, WRAPAROUND = TRUE, FORMSCAPABLE = FALSE, "."
```

```
term line 72
#LINE = 72, PAGE = 24, MAXOUTPUT = 1920, MAXINPUT = 1920, SCREEN,
WAIT, WRAPAROUND = TRUE, FORMSCAPABLE = FALSE, "."
```

### TITLE

The TITLE and CHANGE commands are synonyms. (Refer to the CHANGE command for details.) The only difference between the TITLE and CHANGE commands is that the keyword TO in the <change files> and <change from files> constructs is optional for the TITLE command, but required for the CHANGE command.

# TYPE

## Syntax

```
TYPE <file name> TO <type>
```

## Explanation

The TYPE command changes the FILEKIND attribute of the specified file. If no <file name> is specified, the work file is assumed. In this case, the CANDE command updates the work file if necessary before changing the type. The <type> can be any of the names specified in Section 2, "Basic Constructs," under the heading <type>.

If a <file name> is used, it must not contain a usercode or an asterisk (\*) prefix.

If the location or length of the sequence number field differs between the old file type and the new file type, CANDE considers them incompatible (unless the new file type is DATA or CDATA).

CANDE allows files other than work files to be changed to an incompatible type. However, warning messages are issued.

CANDE allows you to change the work file to an incompatible type as long as you do not attempt to modify the file. Once the work file has been modified, CANDE must update it when a TYPE command is entered. Therefore, you should not attempt to modify a work file following a change to an incompatible type until you are certain that it contains valid sequence numbers in the correct location.

The LIST command can be used to verify that a file actually contains valid sequence numbers in the sequence field.

Note that CANDE might not allow further manipulation of an altered work file that has been changed to an incompatible type. If this occurs, a solution might be to resequence the file with the OVERRIDE option. However, this might result in lost data, depending on the location and/or length of the sequence number field for the new file type.

If the record size of the file or work file is not equal to the default record size for the new type, the TYPE command is accepted, but a warning message is issued.

## Examples

```
type algol
#

WHAT
#WORKFILE WOOF: ALGOL (PREVIOUSLY FORTRAN)
```

## TYPE Command (cont.)

---

TYPE DATAFILE COBOL

#

ty fortran/file seq

#

MAKE SHORT SEQ

#WORKFILE SHORT: SEQ

100abc

200def

TYPE ALGOL

#UPDATING

# WARNING: ACTUAL RECORD SIZE (14 WORDS) IS LESS THAN THE DEFAULT  
(15 WORDS) FOR TYPE ALGOL

#

LIST

100 abc

200 def

#

TYPE COBOL

# WARNING: ORIGINAL SEQUENCE FIELD (FOR TYPE SEQ) WAS COLUMNS 73-80;  
TYPE COBOL EXPECTS SEQUENCE NUMBERS IN COLUMNS 1-6

#

LIST

#NON-DIGIT IN SEQ (1-6) "abc ": 00000100

#FILE:(UZER)CANDE/SHORT ON USERPACK

#

TYPE CSEQ

# WARNING: ORIGINAL SEQUENCE FIELD (FOR TYPE SEQ) WAS COLUMNS 73-80;  
TYPE CSEQ EXPECTS SEQUENCE NUMBERS IN COLUMNS 1-5

#

LIST

#NON-DIGIT IN SEQ (1-5) "abc ": 00000100

#FILE:(UZER)SHORT ON USERPACK

#

TYPE ALGOL

# WARNING: ACTUAL RECORD SIZE (14 WORDS) IS LESS THAN THE DEFAULT (15  
WORDS) FOR TYPE ALGOL

#



# UPDATE

## Syntax

— UPDATE —————|

## Explanation

The UPDATE command forces the system to update the work file immediately. All changes, additions, and deletions accumulated are applied to the work file, and an updated work file is generated. CANDE acknowledges the request by displaying “#UPDATING” and then signifies completion of the update by displaying a number sign (#).

The UPDATE command is invoked implicitly whenever needed to permit another command to function properly; therefore, the UPDATE command is not usually entered on the terminal. The UPDATE command is implicitly invoked when any of the following commands are encountered: BIND, COMPILE, EXCLUDE, EXECUTE, INSERT, MERGE, MOVE, RANGE, REPLACE, RESEQ, RMERGE, RUN, SAVE, START, TYPE, WRITE, and UTILITY.

However, at least three situations require the user to enter the UPDATE command on the terminal. These three cases are as follows:

- The UPDATE can be explicitly invoked to ensure that a work file is a private copy. If user X gets a file from a library other than the library for his usercode (for example, GET (Y)HISFILE), the work file source resides in the library of user Y and is subject to any change or removal done by Y on that file until an UPDATE is invoked (either explicitly or implicitly) by user X. The update action creates a separate copy of the file for user X in a work file area.
- The UPDATE can be explicitly invoked to ensure a complete work file recovery in the event of a system failure. If the system fails during the modification of a work file, the last several changes to the work file (a maximum of four) may be lost. This loss occurs because CANDE defers changes to a work file as long as possible before invoking the time-consuming UPDATE. If the work file is updated when a system failure occurs, the work file is recovered completely.
- The UPDATE can be explicitly invoked to ensure that a complete update of the file is accomplished. That is, the implementation of CANDE is such that if enough changes (none requiring an update) are made to the work file, more than one update may be required to accomplish a complete update of the file. This UPDATE use produces a file that represents all outstanding work file changes. In the case where the work file is either type DATA or CDATA, this process sometimes results in an incorrect update. CANDE discards any outstanding work file changes that cannot be made with one update of the file and displays the following error message:

DATA LOST, LIMITED CHANGES ALLOWED TO DATA FILES

To prevent an incorrect update from occurring, a periodic update of a DATA or CDATA file must be performed using the UPDATE verb.

## UPDATE Command (cont.)

---

For files of type CDATE and DATA, an UPDATE causes CANDE's internal sequence numbers to be recalculated. These numbers are computed by multiplying the relative record number by 100.

### Example

```
UPDATE  
#UPDATING  
#
```

# UTILITY

## Syntax

```

UTILITY [ $ ] <file title> [ <text> ] [ '<text>' ]
; <task equation list>

```

## Explanation

The **UTILITY** command allows user-written programs to modify the current work file. The utility to be used is specified by name (as in executing any other program). An optional string of text can be passed in an array parameter to the utility. **CANDE** makes the work file available to the utility for access or modification, using file equation and the task value for communication. The `<text>` is any string of data not including an unquoted semicolon (;). One or more elements from the `<task equation list>` can be included (as for the **EXECUTE** verb).

The dollar sign (\$), `<file title>`, and `<task equation list>` elements are treated the same as for the **EXECUTE** verb. The `<text>` is copied into an array that is passed as a parameter to the utility. Except between quotation marks, any lowercase letter is translated to uppercase and any redundant blanks are discarded. (A blank is considered redundant if it is adjacent to another blank, or if it either begins or ends the `<text>`.) If any alterations are made to the work file, the work file is updated before the utility is invoked.

The utility must be a procedure accepting an array parameter, as in the following ALGOL heading:

```

$ SET LEVEL 2
PROCEDURE U(A); ARRAY A[*];

```

The array contains the text from the utility command, in EBCDIC, followed by at least one NUL. The length of the array is arbitrarily large; all characters beyond the text are NUL.

The five files shown in Table 3-8 may be file-equated.

**Table 3-8. File-Equatable Files**

Inname	Attribute	Specification
WORKFILE	TITLE	Name of work file
WORKSOURCE	TITLE	Saved or unsaved worksource
NEWWORKSOURCE	TITLE	Potential new unsaved worksource

continued

Table 3-8. File-Equatable Files (cont.)

Intrname	Attribute	Specification
	INTMODE	
	EXTMODE	
	UNITS	
	MAXRECSIZE	
	BLOCKSIZE	
	FILEKIND	
	SECURITYTYPE	
	SECURITYUSE	
	SECURITYGUARD	
	CYCLE	
	VERSION	
	SAVEFACTOR	
WORKOBJECT	TITLE	Saved or unsaved workobject
NEWWORKOBJECT	TITLE	Potential new unsaved workobject

If the CANDE session has no work file, no file equation is done. If a work file exists, it may or may not be named. If a work file exists and is named, then WORKFILE.TITLE is equated to the work file name. The name has neither a usercode prefix nor a family suffix and serves only for identification. The utility is not expected to process any input or output through the file WORKFILE.

A work file can be empty. If the work file is not empty, a worksource file is made containing the line images for the file. If a worksource exists, then WORKSOURCE.TITLE is equated to its title. A utility reads this file to obtain the work file contents. To be consistent with CANDE conventions, the utility should not write to WORKSOURCE.

If the work file has been compiled, the object code is contained in a workobject file. If a workobject exists, WORKOBJECT.TITLE is equated to its title. A utility could read this file to display or analyze the code.

If any work file (named or not, empty or not) exists, file equation is performed for the files NEWWORKSOURCE and NEWWORKOBJECT. All the attributes listed above for NEWWORKSOURCE are equated to the appropriate values for the work file. The utility may write a new worksource with these attributes to replace the old worksource file. The TITLE of NEWWORKOBJECT is equated; a file of this name may be created if the utility invokes (or is) a compiler to create a new workobject.

All the source and object titles are complete, including usercode prefix and family suffix, except that ON DISK cannot appear for files on disk.

On entry to the utility, the utility TASKVALUE is set as follows:

Bit	Value	Condition
[47:43]	0	Not used
[00:01]	1	If a work file exists
[01:01]	1	If a work file name exists
[02:01]	1	If a worksource exists
[03:01]	1	If a workobject exists
[04:01]	1	This bit is always set to 1 by CANDE and prevents the utility from crunching the NEWWORKSOURCE file when it is created. This bit can be set to 0 (zero) by other programs that use the utility program interface.

On termination of the utility, CANDE interprets the TASKVALUE as follows:

Bit	Value	Interpretation
[46:01]	1	An error occurred; CANDE terminates or resumes a SCHEDULE session or holds pending queued input. The utility must generate its own error message as CANDE does not. Bits 23 and 22 remain significant when bit 46 is set.
[22:01]	1	A new worksource has been written, using the title and other attributes supplied for NEWWORKSOURCE.
[23:01]	1	A new workobject has been written, using the title supplied for NEWWORKOBJECT.

If the utility is abnormally terminated, CANDE ignores the TASKVALUE and acts as though bit 46 were ON and all others OFF.

If a work file exists, an UPDATE to that work file is done before the UTILITY command is executed.

### Examples

The following example assumes a CANDE user with usercode of MY and family specification of DISK = MINE OTHERWISE DISK, using station number 12, with access to a utility program named OBJECT/P. The following session fragments result in running the utility with the attributes noted. Those TITLE attributes not mentioned in each case default to the INTNAME. In each case that NEWWORKSOURCE.TITLE is specified, all the other attributes listed previously have been equated to the values appropriate to the work file.

## UTILITY Command (cont.)

---

No work file exists.

```
INPUT
-----
REM;U P

STATUS OF UTILITY TASK
-----
MYSELF.TASKVALUE = 0
```

A named, empty work file exists.

```
INPUT
-----
MAKE T;U P

STATUS OF UTILITY TASK
-----
MYSELF.TASKVALUE = 3
WORKFILE.TITLE = "T."
NEWWORKSOURCE.TITLE = "(MY)CANDE/TEXT120 ON MINE."
NEWWORKOBJECT.TITLE = "(MY)CANDE/CODE120 ON MINE."
```

A named work file with an unsaved worksource and workobject exists.

```
INPUT
-----
MAKE T
100BEGIN
...
900END.
C
U P

STATUS OF UTILITY TASK
-----
MYSELF.TASKVALUE = 15
WORKFILE.TITLE = "T."
WORKSOURCE.TITLE = "(MY)CANDE/TEXT120 ON MINE."
WORKOBJECT.TITLE = "(MY)CANDE/CODE120 ON MINE."
NEWWORKSOURCE.TITLE = "(MY)CANDE/TEXT120 ON MINE."
NEWWORKOBJECT.TITLE = "(MY)CANDE/CODE120 ON MINE."
```

A named work file with a saved worksource and workobject exists.

```

INPUT
-----
GET Z;U P

STATUS OF UTILITY TASK
-----
MYSELF.TASKVALUE = 15
WORKFILE.TITLE = "Z."
WORKSOURCE.TITLE = "(MY)Z ON MINE."
WORKOBJECT.TITLE = "(MY)OBJECT/Z ON MINE."
NEWWORKSOURCE.TITLE = "(MY)CANDE/TEXT12Ø ON MINE."
NEWWORKOBJECT.TITLE = "(MY)CANDE/CODE12Ø ON MINE."

```

A named work file with a borrowed worksource exists.

```

INPUT
-----
G (U)X ON F AS W
U P

STATUS OF UTILITY TASK
-----
WORKSOURCE.TITLE = "(U)X ON F."
NEWWORKSOURCE.TITLE = "(MY)CANDE/TEXT12Ø ON MINE."
NEWWORKOBJECT.TITLE = "(MY)CANDE/CODE12Ø ON MINE."

```

An unnamed work file with a borrowed worksource exists.

```

INPUT
-----
G SYMBOL/ALGOL
U P

STATUS OF UTILITY TASK
-----
WORKSOURCE.TITLE = "*SYMBOL/ALGOL."
NEWWORKSOURCE.TITLE = "(MY)CANDE/TEXT12Ø ON MINE."
NEWWORKOBJECT.TITLE = "(MY)CANDE/CODE12Ø ON MINE."

```

# VOID

### Syntax

— VOID —<sequence number> —  
                                  └<text>┘

### Explanation

The VOID command deletes a record on a page while in page mode. (Refer to “Page Mode Operations” in Section 1, “General Information” for more information.) The act of simply erasing records displayed by CANDE, such as by writing over them with new sequence numbers, does not automatically delete those records when transmitted to CANDE. The VOID command must precede all characters on a line and begin in the first column, followed by the sequence number. Any number of blanks, or none at all, may separate the VOID command and the sequence number. When the VOID command is used, <text> is ignored. Therefore the text of the record being voided does not have to be cleared from the screen.

The VOID command deletes only one record at a time and is used while in page mode. It allows the user to continue entering text in page mode without interruption. The DELETE command can be used to delete one record or a group of records at any time. However, using a DELETE command within a page interrupts page mode processing.

The sequence number is assumed to consist of the maximum number of digits allowed for the sequence field of the work file. (Refer to Table 2-1 for information about the sequence fields of the different file types.) After the VOID or V, CANDE interprets as many consecutive digits as are available, up to the maximum length, as being part of the <sequence number>. This means that if the first digit of the sequence field has been overlaid with a V, CANDE scans the remaining seven digits (for an eight-digit sequence field, such as ALGOL) and continues scanning into the text field for one more digit unless a nonnumeric character is encountered.

In order for CANDE to verify that a voided record is correctly in sequence with other records transmitted on the same screen, the following conditions must be met:

- The sequence numbers of both the preceding and following records must be valid and in correct sequential order with both the voided records and with each other.
- Both the preceding and following records must not be voided.
- The cursor must be placed beyond the record that follows the voided record before the screen is transmitted.

Note that validation is needed only when it is possible for CANDE to misinterpret the sequence number of the voided record. In cases where there is no ambiguity (that is, the text portion does not begin with a digit, and/or the sequence number has sufficient leading zeros), CANDE deletes the correct record.



### Examples

The following examples can each be used to delete the record with the sequence number 1000:

```
V00001000 THIS IS TEXT
V 00001000 THIS IS TEXT
V 1000
VOID 1000THIS IS TEXT
VOID 000010000 THIS IS TEXT
```

In the next example, the maximum sequence field allowed is eight digits. Note that digits in the text field immediately follow the sequence field. Because the *V* overlays a leading zero, the next eight consecutive digits are interpreted by CANDE as being the sequence number of the line to be deleted (although the eighth digit was intended to be part of the text field). As a result, the second line, instead of the first line, is deleted.

Before transmitting the VOID command, the text is as follows:

```
V00001001ST LINE
000010012ND LINE
000050003RD LINE
```

After the VOID command is transmitted, the text would be as follows:

```
000001001ST LINE
000050003RD LINE
```

# WFL

### Syntax

— WFL —<job>—————%

### Explanation

The WFL command invokes the Work Flow Language (WFL) compiler to process the <job>. If the specified <job> construct is free of syntax errors, the resulting job is invoked.

A WFL command with an explicit job heading, such as WFL BEGIN JOB (new WFL) or WFL JOB J; BEGIN (old WFL), is processed according to the specifications in the job heading. Otherwise, the <job> is sent to the new WFL compiler regardless of the option setting. Old WFL is invoked only if the USEOLDWFL option is set and a syntax error is found using new WFL. For example, if the USEOLDWFL option is set, the <job> is sent to the new WFL compiler. If a syntax error is found in new WFL, then old WFL is invoked.

*Note:* The old WFL compiler will be deimplemented on a future release.

Refer to the *WFL Reference Manual* for information about starting a WFL job from a CANDE session.

### Example

```
WFL DISPLAY "STRING";  
#RUNNING 8114  
#BOT 8115 (UZER)WFLCODE  
#8115 DISPLAY: STRING.  
#EOT 8115 (UZER)WFLCODE  
#
```

# WHAT

## Syntax

— WHAT —

## Explanation

The WHAT command indicates the state of the work file. The amount of information provided varies, depending on the current state of the work file; the amount of information relevant or available is indicated. The output may include the following information:

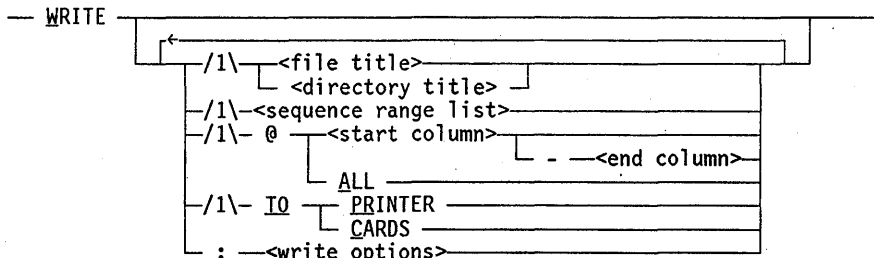
- Title (name)
- Type (filekind)
- Number of records
- Sequence number of last record
- Object file presence
- Saved or unsaved status
- LOCKEDFILE file attribute setting

## Example

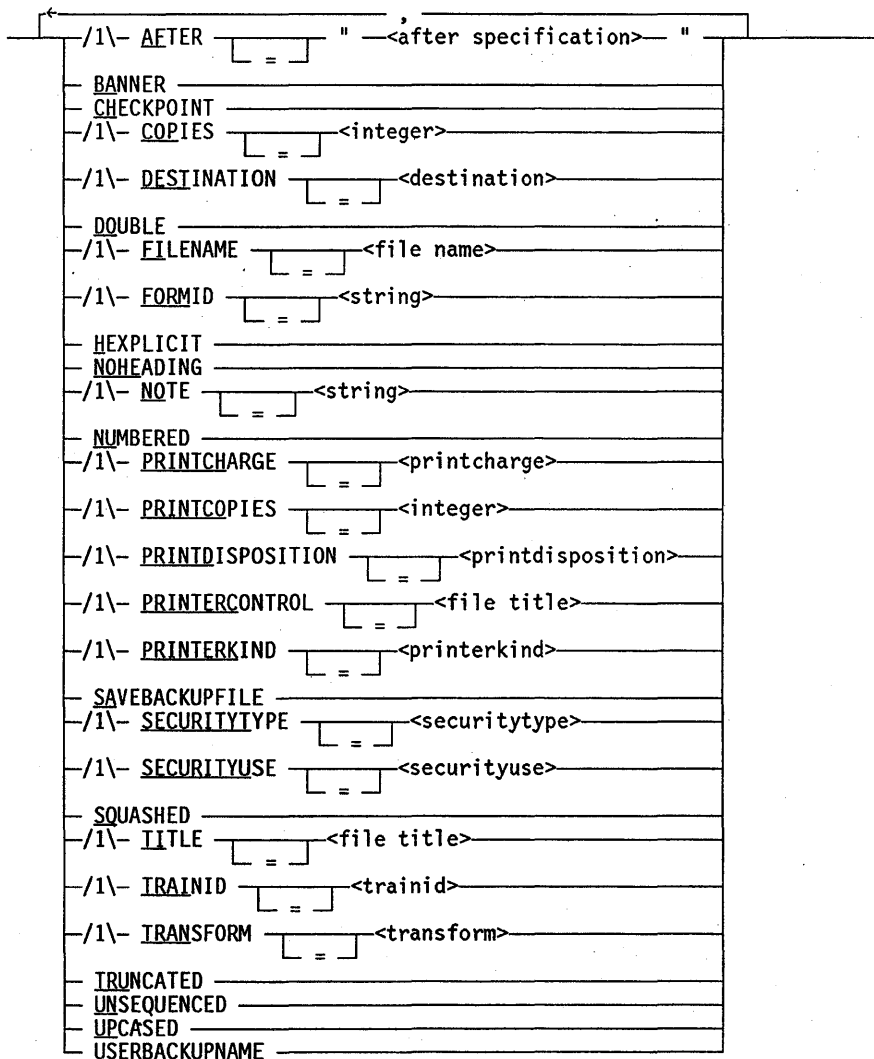
```
WHAT
#WORKFILE PAYROLL/INFO: SEQ, SAVED
LOCKEDFILE=TRUE
```

# WRITE

## Syntax



## <write options>



**<after specification>**

Specifies time in hours and minutes using the form HH:MM. A date can be specified by using the form ON MM/DD/YY. The <after specification> must be within quotation marks (" "). For example, "20:00" represents 8:00 p.m., and "09:30 ON 04/30/86" represents 9:30 a.m. on April 30, 1986. For additional information about the <after specification> string value, refer to the <starttime spec> discussion in the *WFL Reference Manual*. Also refer to the *Print System Guide*.

**<destination>**

Specifies a <device name> in the form <device type> <device number>, where the device type is CP, IP, or LP for a card punch, image printer, or line printer, respectively, and where the device number is the unit number. The destination must appear within quotation marks (" ") and must adhere to the following syntax:

```
— " —<device name> , ————— " —————
           |
           | : <copies> |
```

For example: LP14:2.

To specify a remote device as a destination, use the form STATION <station name>, where <station name> is the name of the remote station. Example: STATION TDS1203LA.

This option can also be used to specify copies for each destination by placing a colon (:) after the device number and ending the statement with an integer representing the number of copies. Each destination specification must be separated from the other by a comma (.). For example, to send four copies to line printer 14 and three copies to image printer 2, enter the following:

```
DESTINATION = "LP14:4, IP2:3"
```

**<printcharge>**

Specifies any valid charge code. The <printcharge> can consist of up to 17 alphanumeric characters and must appear within quotation marks (" ").

```
— " —<printcharge>— " —————
```

**<printdisposition>**

Any one of the mnemonics defined for the PRINTDISPOSITION file attribute in the *Print System Guide*. Examples are DIRECT, EOJ, and DONTPRINT.

**<printerkind>**

Any one of the mnemonics defined for the PRINTERKIND file attribute in the *File Attributes Reference Manual*. Examples are LINEPRINTER, IMAGEPRINTER, and LP.

## WRITE Command (cont.)

---

### <securitytype>

Any one of the mnemonics defined for the SECURITYTYPE file attribute in the *File Attributes Reference Manual*. Examples are PRIVATE, PUBLIC, and GUARDED.

If a security type of CONTROLLED or GUARDED is specified, a guard file name must be entered after the SECURITYTYPE attribute. For additional information about guard files, refer to the SECURITYGUARD attribute discussion in the *File Attributes Reference Manual*.

If the security type is PUBLIC or PRIVATE, the SECURITYUSE attribute can optionally be entered after the SECURITYTYPE attribute. Note that a SECURITYUSE attribute cannot be specified if the security type is GUARDED or CONTROLLED.

Specify the < securitytype > option according to the following syntax:

```
— <securitytype> _____  
    |_____/1\<guard file title>_____  
    |_____/1\<securityuse>_____
```

### <securityuse>

Any one of the mnemonics defined for the SECURITYUSE file attribute in the *File Attributes Reference Manual*. Examples are IN, OUT, and IO.

### <trainid>

Any one of the mnemonics defined for the TRAINID file attribute in the *File Attributes Reference Manual*. Examples are EBCDIC96 and ASCII72.

### <transform>

Specify < transform > according to the following syntax:

```
— " —<transform name> _____ " _____  
    |_____| IN <transform library> _____|
```

If a < transform library > is not specified, the < transform name > must be in the standard transform library.

### Explanation

The WRITE command allows a file (or portions thereof) to be listed on the line printer or punched on cards. As with any output generated by a user, backup files created by using the WRITE command are not actually printed until the user either logs off or uses the SPLIT command.

The <file title> defines the file to be printed or punched. If no <file title> is specified, the work file is assumed. In this case, an UPDATE is done before the WRITE is executed. The <file title> construct can be any file to which the user is allowed access.

If a <directory title> is specified, each file under the given directory is written. However, the WRITE command ignores a file whose file name has more than 11 nodes (excluding the usercode).

The <sequence range list> construct defines the portions of the file to be written. When multiple sequence ranges are specified, a blank record separates each range from its successor. If no <sequence range list> is specified, the entire file is output.

The TO option defines the destination of the output. The PRINTER option directs the output to the site printer, and the CARDS option directs the output to the site card punch. The default, if this option is omitted, is PRINTER.

The @ option specifies the portion of each line to be written. If a <start column> and optional <end column> are specified, then only those columns of each line are written. If ALL is specified, all columns are written adjacent in an unformatted form. If the @ options are omitted, the lines are written in the format described below.

Output files are unlabeled if printed. If a TITLE or FILENAME is provided, both print and punch files are assigned the specified title or file name. Otherwise, punch files are assigned the title of the file or work file being punched; the title appears in an EBCDIC label card at the front of the deck. All non-EBCDIC files are translated to EBCDIC for punching and printing purposes.

Punched files are written in an unformatted form; that is, the punched deck represents a column-by-column image of the file. Lines longer than a card (80 columns) are truncated.

Each line of printer output is formatted as follows:

- The record sequence number is printed in columns 1 through 8, right justified, with leading zeros replaced by blanks (pseudo-sequence numbers are supplied for type DATA or CDATA files). If the sequence field of the record contains nonnumeric, nonblank characters or is less than or equal to the previous record sequence number, \*SEQERR\* is printed instead of the sequence field. In this case, the contents of the sequence field can be found in columns 92 through 99.
- Columns 9 through 12 are blank.

- Type DATA and CDATA files use the remainder of the print line (columns 13 through 132) for the text of the record (unless NUMBERED is specified). If the text field of the file is longer than the remainder of the print line, then squeezing or truncation is performed if the output options so indicate (these options are described in the following paragraphs). If necessary, the text is split and continued on successive lines of output. To use the entire 132-column print line for text, the single output option UNSEQUENCED must be specified.
- For file types other than DATA or CDATA, print columns 13 through 87 are used for the text portion of the record. The text field of type COBOL files (columns 7 through 72) is printed in columns 13 through 78, and the identification field (columns 73 through 78) is printed in columns 81 through 88. For other types of files, the text field is printed in columns 13 through 84 (up to 72 characters per line of printer output). If the text field of the file is longer than 72 characters, then squeezing or truncation can be specified. If necessary, the text is split and continued on successive lines of output.
- Columns 89 through 92 are blank.
- The sequence field of the record is printed in columns 93 through 100. If the file is of type DATA or CDATA, these columns are blank.
- Columns 101 through 102 are blank.
- Columns 103 through 112 are left blank for most types of files; however, for type ALGOL, DCALGOL, and ESPOL files that have patch information in columns 81 through 90, the information is printed in these columns.
- Columns 113 through 114 are blank.
- If the NUMBERED option is specified, the relative record number is printed in columns 115 through 120, prefixed by a number sign (#).
- The remainder of the print line is blank.

The AFTER option defers printing of the backup file until a later time. When specified, the string value of the AFTER option must conform to the WFL <starttime spec>. A null string indicates that the backup file is to be printed as soon as possible.

If the BANNER option is specified, a banner page is printed preceding the text of the backup file. Otherwise, no banner page is printed.

If the NOTE option is specified and the BANNER option has not been specified, the printing of a note is suppressed.

If the BANNER option is specified, the banner page is composed of either the NOTE text if the NOTE text is nonnull, or the file title if the NOTE text is null. The text of the banner page is printed in block characters.

The CHECKPOINT option causes the system to take checkpoints to allow printing to restart in mid-file if the system halt/loads. The interval between checkpoints is determined by the system administrator. If CHECKPOINT is not specified, the system restarts printing from the beginning of the file after a halt/load. If PRINTDISPOSITION = DIRECT, CHECKPOINT is not applicable and is therefore ignored.



The COPIES option specifies the number of times a file is written by CANDE. For example, if a <directory title> is specified and COPIES is set equal to 2, each file in the directory will be written two times. This option is different from the PRINTCOPIES option, which applies to backup files; refer to the explanation of PRINTCOPIES below. The PRINTCOPIES option can be used in conjunction with the COPIES option. If the COPIES option and PRINTCOPIES option are both set to a value greater than 1, the number of copies printed is the value of COPIES multiplied by the value of PRINTCOPIES (COPIES \* PRINTCOPIES).

The DESTINATION option specifies a list of destinations to which the backup file is to be routed. For each device indicated, the system prints the specified number of copies on the device. If the <copies> part of the option is not specified, the number of copies is determined by the PRINTCOPIES option.

If the CANDE command DESTNAME previously specified a station for the session, the destination specified by the DESTINATION option of the WRITE command overrides the DESTNAME value. A warning is displayed when the WRITE command option DESTINATION overrides the DESTNAME value.

The DOUBLE option causes listings to be double-spaced.

The FILENAME option is used in conjunction with the USERBACKUPNAME option to specify a title for the backup file. If the USERBACKUPNAME option is specified, the value assigned to the FILENAME option is used as the title of the backup file. Otherwise, a standard system-generated title is used. If FILENAME is specified and USERBACKUPNAME is not set, the last node of the file title reflects the specified FILENAME. Note that since the TITLE option also affects the naming of the backup file, either FILENAME or TITLE can be specified. However, both should not be specified. If both are specified, the last one specified takes precedence, and the previously specified one is ignored.

The FORMID option, when used with the WRITE command, specifies forms for the printer or card punch by using a string. The string can contain up to 15 characters, and it is displayed on the operator display terminal (ODT). For additional information, refer to the FORMID attribute discussion in the *File Attributes Reference Manual*.

The HEXPLICIT option causes a record that contains nongraphic characters to be printed on two lines. The two characters representing the hexadecimal notation of the nongraphic character appear vertically. The first HEX character appears on the first line. The second HEX character appears on the second line, immediately below the first. Uppercase letters and other graphic characters appear on the first line with a blank on the second line. Lowercase letters are printed in the uppercase equivalent with an underscore ( \_ ). A blank line separates each record. The DOUBLE option is ignored if the HEXPLICIT option is used.

The NOHEADING option suppresses the heading and the two blank lines following the heading. The file is printed starting on the first line of the first page. The heading consists of a blank line followed by a file title, a creation date, and a current time and date.

The NOTE option specifies the message text that is to be printed on the banner page preceding the backup file.

## WRITE Command (cont.)

---

The **NUMBERED** option causes the number of each record to be printed in columns 115 through 120.

The **PRINTCHARGE** option specifies a chargecode, which can consist of up to 17 alphanumeric characters. The print system uses the chargecode to record information for charging printing costs.

The **PRINTCOPIES** option specifies the number of times a backup file is printed by the print system. This option is different from the **COPIES** option, which specifies the number of times a file is written by **CANDE**; refer to the explanation of **COPIES** above for additional information. The **PRINTCOPIES** option can be used in conjunction with the **COPIES** option. If the **COPIES** option and **PRINTCOPIES** option are both set to a value greater than 1, the number of copies printed is the value of **COPIES** multiplied by the value of **PRINTCOPIES** (**COPIES** \* **PRINTCOPIES**).

The **PRINTDISPOSITION** option specifies whether a backup file is to be queued for printing, and if so, when the system should queue the backup file for printing.

The **PRINTERCONTROL** option specifies the printer control file to be associated with the backup file. Note that the **PRINTERCONTROL** specification is not required to be within quotation marks (" ") when used as an option for the **WRITE** command.

The **PRINTERKIND** option specifies the type of output device to which the output is to be directed.

If the **SAVEBACKUPFILE** option is specified, the backup file is saved upon completion of printing. Otherwise, it is removed.

The **SECURITYTYPE** option specifies the security type of the backup file. As is the case with disk files, the default value of the **SECURITYTYPE** attribute is **PRIVATE** for usercoded files and **PUBLIC** for nonusercoded files.

The **SECURITYUSE** option specifies the manner in which a disk file that is protected by security can be accessed by nonprivileged users using nonprivileged programs. **SECURITYUSE** is ignored if a guard file is invoked.

The **SQUASHED** option causes any group of multiple blanks to be reduced to a single blank.

The **TITLE** option is used in conjunction with the **USERBACKUPNAME** option to specify a title for the backup file. If the **USERBACKUPNAME** option is specified, the value assigned to the **TITLE** option is used as the title of the backup file. Otherwise, a standard system-generated title is used. If **TITLE** is specified and **USERBACKUPNAME** is not set, the last node of the file title reflects the file name specified in **TITLE**. Note that since the **FILENAME** option also affects the naming of the backup file, either **FILENAME** or **TITLE** can be specified. However, both should not be specified. If both are specified, the last one specified takes precedence, and the one specified previously is ignored.

The **TRAINID** option specifies the train to be used on the line printer when a file is printed.

The TRANSFORM option specifies a transformation procedure to be executed between the retrieval of records from the source file and the printing of those records on the destination device or devices. The <transform name> is the name of the transform function. The IN <transform library> specification is optional and specifies the library name of which the function is an entry point. The library name must have the correct usercode or family name. If a library name is not specified, the system assumes the function to be an entry point of the standard transform library, PRINTSUPPORT.

The TRUNCATED option causes the text portion of a record that cannot fit in the available space on a line to be truncated to one line and terminated by a slash (/).

The UNSEQUENCED option causes the sequence number listed at the beginning of each line to be suppressed and the record text to begin printing in column 1.

The UPCASED option causes all lowercase letters to be translated to the uppercase equivalent.

The USERBACKUPNAME option controls the default naming conventions for the backup files. When the USERBACKUPNAME option is specified, the backup file is titled using the names specified by either the FILENAME or the TITLE option, applying family substitution as is done for other files. If the USERBACKUPNAME option is not specified, standard system-generated names are used. Note that since both the FILENAME and TITLE options can affect the naming of the backup file, either FILENAME or TITLE can be specified. However, both should not be specified. If both are specified, the last one specified takes precedence, and the one specified previously is ignored.

The options PRINTERCONTROL, TRAINID and PRINTERKIND are not allowed when the CARDS option of the WRITE command has been specified. If any of these three options are selected for a punched file, an error message is generated.

The following file attributes are further discussed in the *File Attributes Reference Manual*:

FORMID  
TITLE  
FILENAME  
SECURITYTYPE  
SECURITYUSE  
TRAINID

The following print file attributes/modifiers are further discussed in the *Print System Guide*:

AFTER	USERBACKUPNAME
DESTINATION	PRINTERCONTROL
NOTE	PRINTCHARGE
TRANSFORM	PRINTERKIND
BANNER	PRINTCOPIES
CHECKPOINT	PRINTDISPOSITION
SAVEBACKUPFILE	

## WRITE Command (cont.)

---

### Examples

```
write  
#RUNNING 8154  
#
```

```
WRI TO PR:DOUBLE, UP FILEID @1-10 100,150-200,END  
#RUNNING 8156  
#
```

## + or -

### Syntax

```
[ + ] [ - ] [ <integer> ]
```

### Explanation

The + and - commands invoke page mode and allow movement through the work file while in page mode. When + or - initiates page mode, a shift is made <integer> records forward (+) or backward (-) from the first record in the file and a page is then displayed. After page mode has been initiated, + and - allow the file to be scrolled by increments of an indicated number of records. The + <integer> construct causes CANDE to display a page <integer> records forward from the first record on the current page. The - <integer> construct causes CANDE to display a page <integer> records backward from the first record on the currently displayed page.

A + or - command can be used to resume page mode after it has been interrupted by a nonpage-invoking command; it provides a page relative to the current page at the time page mode was interrupted. A +5 would provide a page five lines forward from the page that was displayed just before page mode was interrupted.

The default value for <integer> is 1.

A + or - command automatically puts the user into page mode. (Refer to "Page Mode Operations" in Section 1.)

*Next+* appears at the top of the page if + <integer> was just entered. *Next-* appears at the top of the page if - <integer> was just entered.

### Examples

The following example displays the page beginning five records from the top of the currently displayed page:

```
+5
```

The following example displays the page beginning 49 records back from the currently displayed page:

```
-49
```



## Section 4

# CANDE Control Commands

In this section, the CANDE control commands are described and illustrated. These commands provide the ability to control and interrogate the operating environment. The commands must be preceded by the station control character, represented as a question mark (?) in this manual.

Control commands cannot be queued and are performed or rejected immediately. Also, some control commands may be processed in parallel with noncontrol commands.

## ?ASDU

### Syntax

— ? — <mix number list> — ASDU \_\_\_\_\_ %

### Explanation

The ASDU (Actual Segment Descriptor Usage) command displays the number of ASDs that are currently being used for a given job or task. It also displays the maximum number of ASDs used by a job or task for the length of time the job or task is in progress.

### Example

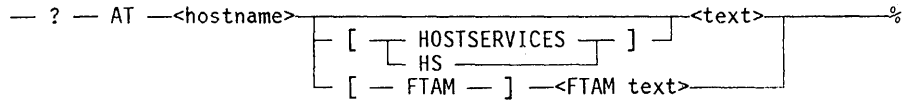
?1234 ASDU

1234 ASD Usage:  
Currently in-use: 80 ASDs  
Maximum in-use: 169 ASDs

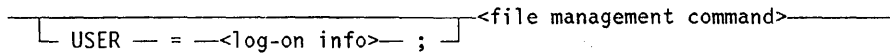


# ?AT

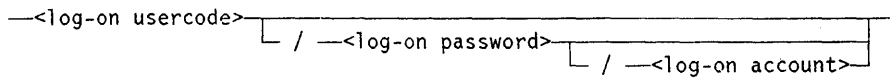
## Syntax



### <FTAM text>



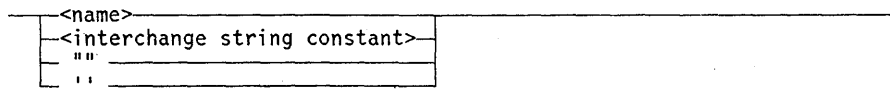
### <log-on info>



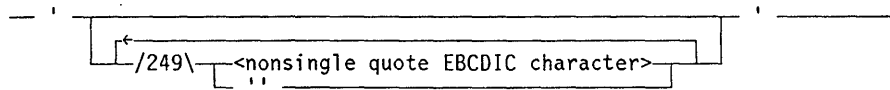
### <log-on usercode>

### <log-on password>

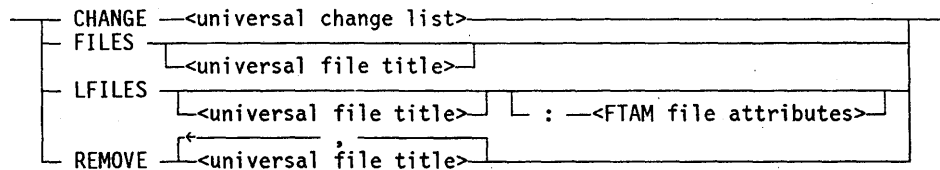
### <log-on account>



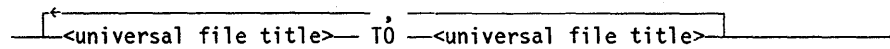
### <interchange string constant>



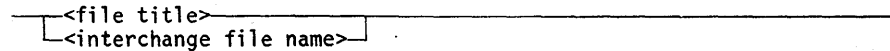
### <file management command>



### <universal change list>



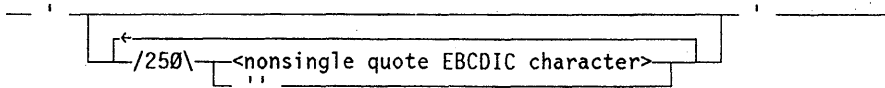
### <universal file title>



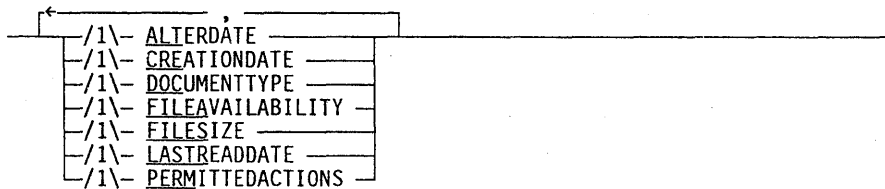
## ?AT Control Command (cont.)

---

<interchange file name>



<FTAM file attributes>



### Explanation

The ?AT command directs information to a remote host identified by the hostname. Additionally, a file access service can be chosen by specification of either Host Services or the File Transfer, Access, and Management (FTAM) service.

If a file access service is specified, then that service is used. However, if the chosen file access service is not supported on both hosts, then an error message that indicates the chosen service is not supported is returned when the ?AT command is attempted.

The FILES and LFILES commands cannot be used with Host Services. If Host Services is specified with either the FILES or LFILES command, then an error message that indicates that the FILES or LFILES command is not recognized is returned.

If a file access service is not specified, then the service that is used is determined by the following:

- If either the FILES or LFILES command is specified, FTAM is used.
- If both hosts support Host Services, then Host Services is used.
- If both hosts support FTAM and a file management command is specified, then FTAM is used.

For more information about FTAM service, refer to the *A Series Distributed System Services (DSS) Operations Guide*.

## Using Host Services

When Host Services is used, the remote host interprets the text as an operator input message. After information has been sent to a remote host through Host Services, a reply from the remote host is received with the following heading:

```
***REPLY FROM <hostname>***
```

If the local hostname is used in the ?AT command, the following text is displayed:

<hostname> IS YOUR LOCAL HOST

The message can then be resubmitted without the AT <hostname> prefix. Chaining of messages is not allowed, as shown in the following example:

```
?AT BLUE AT YELLOW WM
```

## Using the FTAM Service

When FTAM is specified in the ?AT command, a file management command and optional log-on information for the remote host can be passed. The file management command can be one of the following:

- CHANGE
- FILES
- LFILES
- REMOVE

### Using FTAM and the CHANGE Command

The file management command CHANGE can be specified with the ?AT command to rename the file names of disk files on a remote host. FTAM must be supported on both the initiating host and the responding host for successful file name changes. Several files can be listed for file name changes.

### Using FTAM and the FILES Command

The file management command FILES can be specified with the ?AT command to get information about disk files on a remote host. FTAM must be supported on both the initiating host and the responding host for a successful disk file query.

If a disk file name is specified with the FILES command, information for both the file and the directory on the A Series host are sought. If a file name is not specified, the file name and document type for the files of the responding host are returned. File security measures in place at the remote host are considered before queries on individual files or directories are allowed.

### Using FTAM and the LFILES Command

The file management command LFILES can be specified with the ?AT command to query the FTAM file attributes of the disk files on a remote host. If a file name is specified, both the file and the directory on the remote A Series host is queried. If a file name is not specified, all of the files on a remote A Series host are queried. File attribute information from non-A Series host files may not be returned, depending on the user's privileges and security measures in place.

## ?AT Control Command (cont.)

---

FTAM file attributes can be specified; however, if a requested file attribute is not supported by the remote host, the following message is returned to the initiating user for each unsupported file attribute:

NOT SUPPORTED

If FTAM attributes are not specified, then information for all the supported FTAM file attributes is listed, and the unsupported FTAM file attributes are returned with the message NOT SUPPORTED.

If DOCUMENTTYPE is requested, either explicitly or implicitly, the FTAM document-type parameters that are returned with the document type are displayed. When displayed, the document-type parameters are hyphenated in order to distinguish them from attributes that are explicitly requested.

The document-type parameters and the document types that apply when requesting DOCUMENTTYPE follow:

Document-type Parameter	Document Type
MAX-STRING-LENGTH	FTAM1, FTAM2, FTAM3
STRING-SIGNIFICANCE	FTAM1, FTAM2, FTAM3
UNIVERSAL-CLASS-NUMBER	FTAM1, FTAM2

If the document type is not supported by the DSSUPPORT library, the document-type parameters are not displayed.

The following table shows the FTAM file attributes that you can specify with LFILES in the ?AT command. A brief description accompanies each attribute. For more specific information on an attribute, refer to the *A Series File Attributes Programming Reference Manual*.

File Attribute	Meaning
ALTERDATE	Indicates the date and time of the close operation corresponding to the last time that a file was altered.
CREATIONDATE	Indicates the date and time when a file was first opened for creation.
DOCUMENTTYPE	Indicates the data types of the contents of the file and the structuring information.
FILESIZE	Indicates the length of the file in octets.
FILEAVAILABILITY	Indicates whether a delay should be expected during file access.
LASTREADDATE	Indicates the date and time when the file was last read.
PERMITTEDACTIONS	Indicates the set of actions that can be performed on the file. The PERMITTEDACTIONS attribute has meaning only for FTAM file services. This attribute is ignored for file accesses not using FTAM.

## Using FTAM and the REMOVE Command

The file management command REMOVE can be specified with the ?AT command to delete the disk files on a remote host. FTAM must be supported on both the initiating host and the responding host for successful file deletions. Several files can be listed for file deletions.

## Examples

The following examples show various aspects of using the ?AT command.

### Example 1

The following example shows the ?AT command syntax using Host Services:

```
?AT BLUE [HOSTSERVICES] J

*** REPLY FROM BLUE ***
----- JOB STRUCTURE -----
2 * 1566 JOB 50 T
1 * W..1568 50 SYSTEM/CARDLINE ON SYSPACK
```

In the following example, the system does not respond after the FA TITLE command executes:

```
?AT BLUE 1568 FA TITLE=A ON DISK

*** REPLY FROM BLUE ***
NO RESPONSE GENERATED
```

### Example 2

The following example uses the FTAM service and the file management command CHANGE to rename a file at the host OSIHOST2. No usercode, password, or chargecode is passed to the remote host.

In this example, OSIHOST2 is a remote host that runs the MS-DOS® operating system. Both hosts support the FTAM service.

```
?AT OSIHOST2 [FTAM] USER = "";
CHANGE 'c:\ftam\design.doc' TO 'c:\ftam\design.bak'

*** RESPONSE FROM OSIHOST2 ***

'C:\FTAM\DESIGN.DOC' CHANGED TO 'C:\FTAM\DESIGN.BAK'
```

**Example 3**

The following example uses the FTAM service and the file management command REMOVE to delete a file at the remote personal computer host OSIHOST2.

In this example, OSIHOST2 is a remote host that runs the MS-DOS operating system. Both hosts support the FTAM service.

```
?AT OSIHOST2 REMOVE 'c:\ftam\design.doc'  
  
*** RESPONSE FROM OSIHOST2 ***  
  
'C:\FTAM\DESIGN.DOC' REMOVED
```

**Example 4**

The following example uses the FTAM service and the file management command FILES to inquire about a file named A/B at the host OSIHOST1. The usercode LEDGER and password GENERAL are passed to the remote host. OSIHOST1 is an A Series remote host and supports the FTAM service.

```
?AT OSIHOST1 [FTAM] USER = LEDGER/GENERAL ; FILES A/B  
  
*** RESPONSE FROM OSIHOST1 ***  
  
(LEDGER)A/B ON DISK : FTAM2  
(LEDGER)A/B/C ON DISK : FTAM3
```

The following example uses the FTAM service and the file management command FILES to inquire about all of the files on the host OSIHOST2. No usercode, password, or chargecode is passed. The remote host OSIHOST2 runs the MS-DOS operating system and supports the FTAM service.

```
?AT OSIHOST2 USER = ""; FILES  
  
*** RESPONSE FROM OSIHOST2 ***  
  
'C:\FTAM\DESIGN.DOC' : FTAM2  
'C:\WORD.COM' : FTAM3
```

The following example uses the FTAM service and the file management command FILES to inquire about the file 'c\ftam\design.doc' on the host OSIHOST2. The usercode TECH and password ENTER are passed to the remote host. The remote host OSIHOST2 runs the MS-DOS operating system and supports the FTAM service.

```
?AT OSIHOST2 [FTAM] USER = TECH/ENTER ;FILES 'c:\ftam\design.doc'  
  
*** RESPONSE FROM OSIHOST2 ***  
  
'C:\FTAM\DESIGN.DOC' : FTAM2
```

**Example 5**

The following example uses the FTAM service and the file management command LFILES to inquire about the files at the remote A Series host. Both A Series hosts support the FTAM service.

The LFILES command queries the FTAM file attributes of the file A/B on the remote host OSIHST1. The usercode LEDGER and the password GENERAL are passed to the remote host.

For those FTAM attributes that display a date and time stamp, the time difference (either plus or minus) from the Universal Time (UT) is also displayed if it is returned by the remote host.

```
?AT OSIHST1 USER = LEDGER/GENERAL ; LFILES A/B

*** RESPONSE FROM OSIHST1 ***

(LEDGER)A/B ON DISK : FTAM2
  CREATIONDATE           = JAN 29, 1991 @ 16:23:15 [-8:00 FROM UT]
  ALTERDATE              = JAN 30, 1991 @ 15:00:00 [-8:00 FROM UT]
  LASTREADDATE           = NO VALUE AVAILABLE
  FILESIZE               = 928 BYTES
  PERMITTEDACTIONS       = READ/INSERT/ERASE/READ ATTRIBUTE
                        CHANGE ATTRIBUTE/DELETE/
                        TRAVERSAL/REVERSE TRAVERSAL/
                        RANDOM ORDER
  FILEAVAILABILITY        = IMMEDIATE
  MAX-STRING-LENGTH      = 132 BYTES
  STRING-SIGNIFICANCE    = NOT-SIGNIFICANT
  UNIVERSAL-CLASS-NUMBER = IA5STRING
```

The following example uses the LFILES command to query the FTAM file attributes for the file 'c:/june/payroll' at the remote host. OSIHST3 is the remote host that runs the UNIX® operating system. The usercode PAYROLL and the password EMPLOY7 are passed to the remote host.

For those FTAM attributes that display a date and time stamp, the time difference (either plus or minus) from the Universal Time (UT) is also displayed if it is returned by the remote host.

## ?AT Control Command (cont.)

---

```
?AT OSIH0ST3 USER = PAYROLL/EMPLOY7 ; LFILES 'c:/june/payroll'
```

```
*** RESPONSE FROM OSIH0ST3 ***
```

```
'C:/JUNE/PAYROLL' : FTAM3
```

```
CREATIONDATE      = JAN 2, 1991 @ 17:17:34  [-8:00 FROM UT]
ALTERDATE         = JAN 7, 1991 @ 15:12:14  [-8:00 FROM UT]
LASTREADDATE      = NO VALUE AVAILABLE
FILESIZE          = 728 BYTES
PERMITTEDACTIONS  = READ/REPLACE/EXTEND/ERASE/
                  READ ATTRIBUTE/CHANGE ATTRIBUTE/
                  DELETE/
FILEAVAILABILITY  = IMMEDIATE
MAX-STRING-LENGTH = 132 BYTES
STRING-SIGNIFICANCE = NOT-SIGNIFICANT
```

The following example uses the LFILES command to query the FTAM file attribute CREATIONDATE for the file A/B at the remote A Series host OSIH0ST1:

```
?AT OSIH0ST1 [FTAM] USER = ""; LFILES A/B : CRE
```

```
*** RESPONSE FROM OSIH0ST1 ***
```

```
*A/B ON DISK : FTAM2
```

```
CREATIONDATE      = JAN 29, 1991 @ 18:20:14  [-8:00 FROM UT]
```

The following example uses the LFILES command to query the FTAM file attributes CREATIONDATE and FILEAVAILABILITY for the file 'c:\ftam\design.doc' at the remote host. The remote host OSIH0ST2 runs the MS-DOS operating system and supports the FTAM service. The usercode TECH and password ENTER are passed to the remote host.

```
?AT OSIH0ST2 USER = TECH/ENTER; LFILES 'c:\ftam\design.doc':CRE, FILEA
```

```
*** RESPONSE FROM OSIH0ST2 ***
```

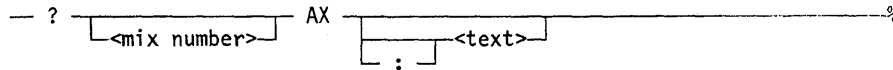
```
'C:\FTAM\DESIGN.DOC' : FTAM1
```

```
CREATIONDATE      = JAN 29, 1991 @ 16:23:15  [-8:00 FROM UT]
FILEAVAILABILITY  = IMMEDIATE
```



# ?AX

## Syntax



## Explanation

The ?AX command passes text to a running program either in response to, or in anticipation of, an ACCEPT statement. If a <mix number> is not specified, the currently running program is assumed. If the colon is used, leading blanks are not removed from the <text>. If the colon is omitted, leading blanks are removed before the <text> is presented to the program.

## Example

```
GET DON1
#WORKFILE DON1: ALGOL,4 RECORDS,SAVED
#OBJECT FILE PRESENT, SAVED
```

```
LIST
100 BEGIN
200 STRING A;
300 ACCEPT(A);
400 END.
#
```

```
RUN
#RUNNING 0208
#0208 ACCEPT:.
```

```
?AX THIS IS A TEST
#0208 GOING
#ET=54.0 PT=0.0 IO=0.1
```

## **?BREAKPOINT**

### **Syntax**

— ? — <mix number> — BREAKPOINT  
BP — %

### **Explanation**

The ?BREAKPOINT command can be used only during a test session of the Test and Debug System. If used when running a program which is not a test session, a warning is produced, and the command has no effect. If the <mix number> is omitted and the station is not executing a program, the response is “#NO ACTIVE TASK.”

When used during a test session, the program being tested is suspended, and control is transferred to the Test and Debug System. For more information, refer to the A Series Test and Debug System (TADS) programming guide that addresses the language of the program being tested.

# ?C

## Syntax

— ? <LSN>  
\* C ————— %

## Explanation

The ?C command lists recently completed jobs and tasks in the mix.

The syntax and results of this command are controlled to some extent by the CANDE *ALLMSG* option. The state of the *ALLMSG* option is controlled by the system operator.

If *ALLMSG* is RESET, a syntax error is issued if an <LSN> or an asterisk (\*) is specified. The only information returned pertains to those jobs or tasks originating from the <LSN> associated with the requestor's usercode.

If *ALLMSG* is SET, information about any <LSN> may be requested. The \* syntax is a shorthand method for specifying the <LSN> to which the user is signed on. If no <LSN> is supplied, all information pertaining to jobs or tasks for the user making the request, regardless of origination point, is returned.

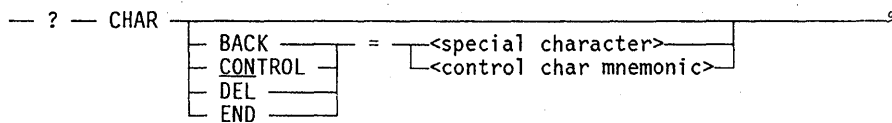
## Example

The asterisk (\*) in the following example specifies that the entry was completed after the previous display of completed entries on the ODT.

```
?C  
----- COMPLETED ENTRIES -----  
*1234/5678 EOJ JOB BACKUP  
1234/5679 EOT SYSTEM/BACKUP ON PACK  
1234/5680 SNTX ALGOL TEST/PROG  
1234/5681 F-DS GEN/TEST/PROG
```

## ?CHAR

### Syntax



### Explanation

The ?CHAR command allows four of the characters reserved for special use at a specific terminal to be redefined. The dynamically alterable characters that can be changed during a session and the corresponding Teletype® default character for each character follow:

Characters	Teletype Default
Backspace	Left arrow
Control	? (Question mark)
Line Delete	DEL (Rubout)
End-of-Text	CR (Return)

The BACK, CONTROL, DEL, and END options cause redefinition of the terminal backspace, control, line delete, and end-of-text characters, respectively. The ?CHAR command cannot be used to redefine the BACK, DEL, and END characters for a pseudostation. If this is attempted, the following error message is displayed:

```
?CHAR CANNOT REDEFINE BACK, DEL, AND END FOR PSEUDOSTATION.
```

The current state of these four characters can be interrogated by entering ?CHAR without additional text. Characters are represented in the response by graphics, mnemonic names (if graphic representation is not appropriate), or two-digit hexadecimal notation (if no mnemonic exists). Lowercase letters are represented by the uppercase equivalents preceded by a slash.

These reserved characters revert to the NDLLI default specifications at log-off and halt/load time.

**Examples**

?CHAR

#BACK = NUL, CONTROL = ?, DEL = DEL, END = CR

?CHAR CON=\*

#BACK = NUL, CONTROL = \*, DEL = DEL, END = CR

?CHAR BACK=BS

#BACK = BS, CONTROL = \*, DEL = DEL, END = CR

## ?% (Comment)

### Syntax

-- ? -- % --<text>-----%

### Explanation

The COMMENT command treats the <text> as a comment and is intended to be useful in schedule files.

### Example

```
?% this is a comment
```

## ?CONNECT

### Syntax

— ? — CONNECT — TO —<hostname> — : —<MCS name> —

### Explanation

The ?CONNECT command allows a user to transfer control of a data comm station from the MCS of a local host to an MCS on a remote host. The ?CONNECT command is the control command version of the CONNECT command and has identical syntax, except for the added defined control character, a question mark (?), in the control command. The defined control character allows the command to be transmitted when the station is not logged on to the local CANDE.

The explanation of the ?CONNECT command is identical to CONNECT (refer to the CONNECT command in Section 3, “CANDE Commands”) with the addition of the following information.

If a command is in progress when the ?CONNECT command is entered, the following message is displayed:

```
#STATION IS BUSY
```

If a remote file is open at the station, the following message is displayed:

```
#STATION IN USE
```

If the station is busy or in use, the command is discarded.

### Example

```
?connect to A15B
#END SESSION 6237 ET=22.5 PT=0.6 IO=0.8
#USER = UZER CHARGE = 9999. 10:52:12 09/06/89
A12:4 SYSTEM/STATION/TRANSFER; HOSTNAME = A12AB
INITIATING DIALOG WITH A15B
DIALOG INITIATED. INITIATING CONNECT.
```

## **?COUNTS**

### **Syntax**

— ? — COUNTS ————— %

### **Explanation**

The ?COUNTS command displays information about the current activity level of CANDE. The response includes the number of active tasks, the number of active compiles, the number of active workers (those doing a file update or text editing command), the number of active stations, and the number of attached stations.

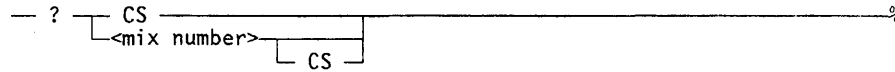
### **Example**

```
?COUNTS  
# 7 TASKS, 5 COMPILES, 1 WORKER; 33 STATIONS ACTIVE, 78 ATTACHED
```



# ?CS

## Syntax



## Explanation

The ?CS command performs one of two functions, depending on the syntax that is used and the current state of the session. One function is to inquire about the status of a compilation task, and the other is to inquire about the system supervisor.

If the ?CS command is issued without a mix number and the session currently has no active task, then the name of the current system supervisor program is displayed.

If the ?CS command is issued with a mix number, then the sequence number of the program that the compiler is currently processing and the number of syntax errors encountered so far are displayed. The compilation task status is also displayed when the ?CS command is issued without a mix number but the session currently has an active task.

## Example

```
?1154 CS  
SEQ = 12345678, NO ERRORS  
  
?STA  
#16:59 NOT BUSY  
  
?CS  
SUPERVISOR: SYSTEM/SUP ON CONTROL
```

## ?CU Control Command

---

### ?CU

#### Syntax

— ? — <mix number> — CU — %

#### Explanation

The ?CU command returns the core utilization for a task. If a <mix number> is not specified, the currently running program is assumed.

#### Example

```
?CU
7228 CORE:  TOTAL  SAVE
          STACK   5191  3787
          SUBSPACE 13860
```

## ?DENY

### Syntax

— ? — DENY ————— %

### Explanation

The ?DENY command denies program requests for I/O from the station. An end-of-file notice is sent to all programs that have files pending, postponed, or open at the station.

The ?END and ?DENY commands differ in that ?END causes end-of-file action only on the input or I/O file. ?DENY also acts on output files.

### Example

```
L DON1
1100 BEGIN
1200 STRING A;
1300 INTEGER I;
1400 FILE DCOM(KIND=REMOTE,MYUSE=OUT);
1500 A:= "HELLO THERE";
1600 DO BEGIN
1700     WRITE (DCOM,20,A);
1800     WAIT ((3));
1900     I:=I+1;
2000     END
2100 UNTIL I EQL 3
2200 END.
#
RUN
#RUNNING 0225
HELLO THERE

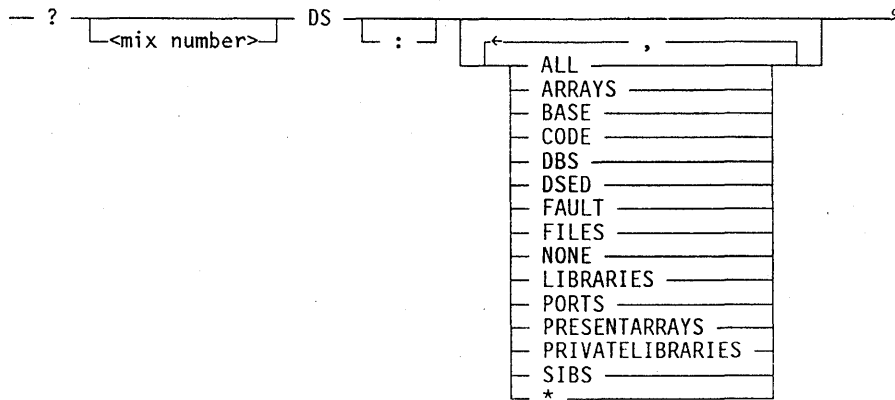
?deny
#
#0225 EOF NO LABEL:DCOM @ 003:0000:1
#I-DS @00001700
#ET=12.7 PT=0.1 IO=0.07
R
#RUNNING 5326
HELLO THERE
?END
#
HELLO THERE
HELLO THERE
#ET=21.8 PT=0.1 IO=0.1
```

## ?DS Control Command

---

### ?DS

#### Syntax



#### Explanation

The ?DS command discontinues the referenced task. If a <mix number> is not specified, the current CANDE command in progress or the currently running task is terminated. If any program dump options are specified, a program dump is taken prior to termination of the program.

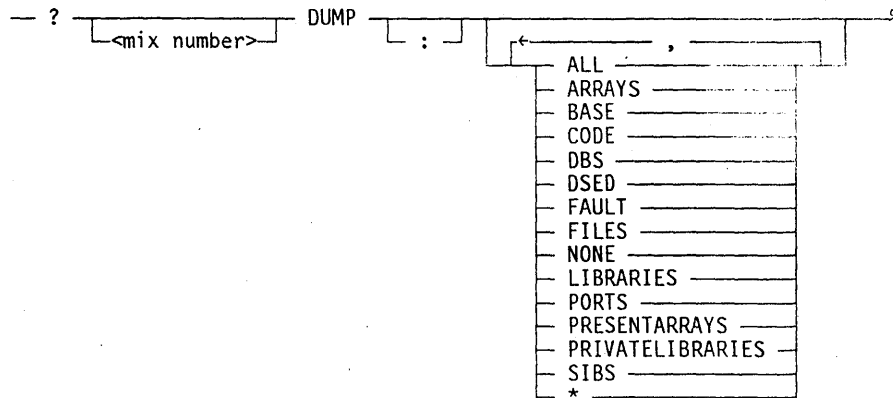
The ?DS command can also be used to discontinue CANDE functions such as LIST, FIND, RESEQ, WRITE, and so forth.

#### Example

```
?DS
#8228 OPERATOR DSED @ 24B:02CF:3*
#0-DS @ 44835200,44904800,00067600,00195600.
#ET=8.7 PT=0.1 IO=0.2
```

# ?DUMP

## Syntax



## Explanation

The ?DUMP command initiates a program dump for the specified task. If a <mix number> is not specified, the currently running task is assumed.

If the FAULT or DSED option is either explicitly or implicitly SET (by a ?DUMP:ALL), a program dump occurs only if the program terminates with a fault or is discontinued, respectively. A dump is not taken when the command is entered.

## Examples

?DUMP ALL

?DUMP ARRAYS

?DUMP



## ?EDIT

### Syntax

— ? — EDIT —<edit specs>—————%

### Explanation

The ?EDIT command allows editing of saved text. The saved text is displayed after it is edited. The EDIT command cannot be used while the station is busy or in sequence mode.

Refer to the FIX command in Section 3, “CANDE Commands,” for an explanation of the <edit specs> construct.

### Example

```
100 this is an example.
```

```
?edit /an/a good  
100 this is a good example.*  
#
```

```
?repeat  
#
```

```
list  
100 this is a good example.
```

## ?END Control Command

---

### ?END

#### Syntax

— ? — END \_\_\_\_\_%

#### Explanation

The ?END command signals end-of-file action to the program that has an input or I/O remote file open.

The ?END and ?DENY commands differ in that ?END causes end-of-file action only on the input or I/O file. ?DENY also acts on output files.

#### Example

```
GET DON1
#WORKFILE DON1: ALGOL,4 RECORDS,SAVED
#OBJECT FILE PRESENT,SAVED
```

```
LIST
100 BEGIN
200 STRING A;
300 ACCEPT(A);
400 END.
#
```

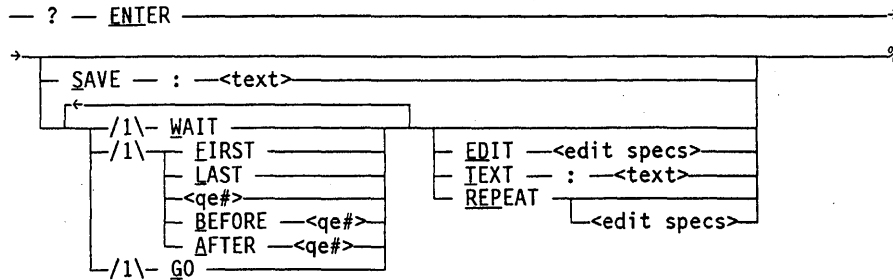
```
run
#RUNNING 0261
0261 ACCEPT:.
```

```
?end
#
```



# ?ENTER

## Syntax



<qe#>

-----<integer>----->

## Explanation

The ?ENTER command allows new entries to be placed in the visible portion of the queue or an entry to be made in the saved text queue.

If the SAVE option is specified, the <text> following the colon is entered in the saved text queue. If the state of the queued input is pending, the state of the queued input is changed to waiting.

If the SAVE option is not specified, an entry is made in the queue. If no queue entry number (<qe#>) is supplied or FIRST is specified, the new entry is placed before the entry that is presently first in the queue. The LAST option places the new entry after the entry that is presently last in the queue (possibly overriding visible restrictions). If <qe#> is supplied, the new entry is put BEFORE, AFTER, or in the place of the queue entry number specified. The state of the queued input is changed to waiting.

The WAIT option changes the user's queued input state to waiting when that entry becomes the first one in the queue.

The TEXT option causes the <text> following the colon to be the text of the new queue entry; otherwise, the user's saved text is the text of the new queue entry.

The EDIT option indicates that the saved text is to be edited and displayed before being entered in the queue.

The GO option causes the session to resume with the first entry in the queue. GO must not be followed by REPEAT.

The REPEAT option causes the session to resume with the saved text. If editing specifications follow the REPEAT option, the saved text is edited and displayed before the REPEAT option is processed. Refer to the FIX command in Section 3, "CANDE Commands," for an explanation of <edit specs>.

## ?ENTER Control Command (cont.)

---

The ?ENTER command cannot be used while the station is busy or in single-line sequencing mode or page mode.

### Examples

```
E
#UPDATING
#COMPILING 5361

SA; G NEXT
#QUEUED
  WAIT(3);
00000740 ERROR-ILLEGAL EVENT DESIGNATOR
#SNTX
#ET=9.2 PT=0.4 IO=0.3
#QUEUED INPUT PENDING

?SH Q
  (1)SA; G NEXT

?WAIT
#

?ENTER BEFORE 1 TEXT: FIX 740/3/(3)
#

?ENTER AFTER 1 TEXT:E
#

?SH Q
  (1) FIX 740/3/(3)
  (2)E
  (3)SA; G NEXT

?GO
#
#UPDATING
#COMPILING 5371
#ET=6.6 PT=0.1 IO=0.1
#
#WORKFILES TEST SAVED
#NO FILE: NEXT
#
```

## ?EOL

### Syntax

— ? — EOL — %  
└─<special character>─┘  
└─<control char mnemonic>─┘

### Explanation

The ?EOL command is valid only from a schedule station and is used to change the end-of-line character for a schedule session. The <special character> is the new end-of-line character. If the <special character> is not specified, the EOL feature is disabled until the next ?EOL command. This command overrides the EOL option used in the SCHEDULE command for all lines subsequent to the ?EOL command. Refer to the SCHEDULE command in Section 3, “CANDE Commands,” for more information on how to use this feature.

### Examples

```
?EOL #  
#EOL IS #
```

```
?EOL  
#EOL DISABLED
```

## ?FA Control Command

---

### ?FA

#### Syntax

```
-- ? [ <mix number> ] FA --<file attribute list> %
```

#### Explanation

The ?FA command allows file attributes for referenced tasks waiting on an RSVP message to be changed. If a <mix number> is not specified, the currently running program is assumed. Refer to the *File Attributes Reference Manual* for additional information about file attributes.

#### Example

```
E  
#RUNNING 1234  
#1234 NO FILE X ON DISK ELSE PACK(DK)  
  
?FA TITLE=NEW/FILE  
#GOING
```

## ?FR

### Syntax

-- ? <mix number> FR \_\_\_\_\_ %

### Explanation

The ?FR command specifies that the input tape reel just read by the task is the final reel in an unlabeled tape file. The command can also be used to respond to a "#RECOPY REQD" message from LIBRARY/MAINTENANCE. In this case, further use of the tape unit on which the copy error occurred is discontinued. If a <mix number> is not specified, the currently running task is assumed.

# ?GO

### Syntax

— ? — GO \_\_\_\_\_%

### Explanation

The ?GO command can be used only when the state of the user's queued input is waiting or pending. This command causes the session to be resumed with the first entry in the queue. The first entry is always removed and performed. The state of the user's queued input is set to normal by the ?GO command.

The ?GO command cannot be used while the station is busy.

### Example

```
run;l
#RUNNING 0395
0395 ACCEPT:.
```

```
?ds
#0395 OPERATOR DSED @ 003:0002:5*
#O-DS @ 00000300
#ET=2:34.0 PT=0.1 IO=0.1
#QUEUED INPUT PENDING
```

```
?go
#
100 BEGIN
200 STRING A;
300 ACCEPT(A);
400 END.
```



## ?HI Control Command (cont.)

---

The following is an example of retrieving information about the progress of a file transfer from a user on host HOSTA:

```
COPY (ABC)FILE/1 AS (ABC)F1 FROM ICE(PACK) TO DISK(HOSTNAME=HOSTB)
```

```
#RUNNING 1231
```

```
#BOT 1232 (ABC)WFLCODE
```

```
#BOT 1233 *FILE/TRANSFER
```

```
?1233 HI
```

```
#1233 COPYING (ABC)FILE/1 ON ICE AS (ABC)F1 ON DISK FROM HOSTA TO HOSTB
```

```
#1233 ELAPSED TRANSFER TIME 00:01 = 7% TRANSFERRED
```



## ?HN

### Syntax

— ? — HN ————— %

### Explanation

The ?HN command displays the current hostname of the host, as well as the hostnames of any other BNA hosts available for port dialogs or host services functions.

Each host in the network is identified by a hostname that is used to identify the location of resources to which access is desired.

### Example

```
?HN
HOSTNAME: BLUE
USABLE BNA HOST ARE:
```

PINK	GREEN	ORANGE	PURPLE
SYS36	HN68AB	TESTSYS	HOSTY
YELLOW	WHITE	RED	BLACK

## ?ID Control Command

---

### ?ID

#### Syntax

— ? — ID \_\_\_\_\_%

#### Explanation

The ?ID command displays the current DCPREFIX and the future DCPREFIX (if any). The DCPREFIX is used by the system to determine which NIF and DCPCODE files are to be used by the data comm subsystem.

#### Examples

```
?ID  
NIF: DC/BLUE/Ø
```

```
?ID  
NIF: SITENDL ON PACK  
NIF TO BE: TESTNDL
```

## ?JA

### Syntax

— ? <LSN>  
\* JA \_\_\_\_\_ %

### Explanation

The ?JA command displays all jobs and tasks running under the usercode of the station. The information about each job or task may include the following:

- Waiting, scheduled, or error status
- Job number
- Mix number
- Priority
- Task name
- Compiler name

The syntax and results of this command are controlled to some extent by the CANDE ALLMSG option. The state of the ALLMSG option is controlled by the system operator.

If ALLMSG is RESET, a syntax error is issued if <LSN> or asterisk (\*) is specified. The only information returned pertains to those jobs or tasks originating from the user's <LSN> with which the <usercode> is associated.

If ALLMSG is SET, information about any <LSN> can be requested. The asterisk (\*) syntax is a shorthand method for specifying the <LSN> to which the user is signed on. If no <LSN> is supplied, all information pertaining to jobs or tasks for the user making the request, regardless of origination point, is returned.

### Example

```
?JA
----- JOB STRUCTURE -----
5812 JOB 50 RUN MY/PROG ON SIXPACK
W..5820 55 MY/PROG ON SIXPACK
```

## ?LIB Control Command

---

### ?LIBS

#### Syntax

— ? -<LSN>  
\* LIBS \_\_\_\_\_%

#### Explanation

The ?LIBS command displays a list of the frozen libraries under the usercode of the station. The information about each library includes the following:

- Mix number
- Library name
- Number of users

The syntax and results of this command are controlled to some extent by the CANDE *ALLMSG* option, the value of which is assigned by the CANDE network control command ?OP (refer to the *A Series CANDE Configuration Reference Manual* for information about ?OP).

If *ALLMSG* is RESET, a syntax error is issued if <LSN> or asterisk (\*) is specified. The only information returned pertains to those libraries with the user's <usercode> that originate from the user's <LSN>.

If *ALLMSG* is SET, information about any <LSN> can be requested. The asterisk (\*) is a shorthand method for specifying the <LSN> to which the user is signed on. If no <LSN> is specified, all information pertaining to libraries for the user making the request, regardless of origination point, is returned.

Note that the system libraries accessed by user programs are not visible through the ?LIBS command. The only visible libraries are those that were associated with a user's <usercode> when they were frozen. For those libraries visible through the ?LIBS command, other relevant CANDE control commands such as ?DS, ?TI, ?Y, ?OT, and ?CU are allowed. The system command THAW is also allowed. The CANDE control command ?AX can be used after the library is thawed.

#### Example

```
?LIBS
----- 2 FROZEN LIBRARIES -----
* 7216 JOB (1) (SMITH)OBJECT/LIB on PACK1
* 7232 JOB (2) (SMITH)OBJECT/LIB2 on PACK2
```

In the preceding example, the number enclosed in parentheses indicates the number of users of that library.

## ?MCS

### Syntax

— ? — MCS —<MCS name>—————%

<MCS name>

—<file title>—————|

### Explanation

The ?MCS command causes control of the station to be transferred to another MCS. The <MCS name> must designate an NDLLI-declared MCS. If an unsaved work file exists, the following message is displayed:

#REMOVE OR SAVE WORKFILE

The unsaved work file must be removed or saved, and the ?MCS command must be re-entered before the station can be transferred.

If a CANDE command is in progress, the following message is displayed:

#STATION IS BUSY

If a remote file is open at the station, the following message is displayed:

#STATION IN USE

If the station is busy or in use, the command is discarded.

The ?MCS command is the same as the MCS command, except that the ?MCS form, as a control command, can be entered while the station is not logged on.

## ?MM Control Command

---

### ?MM

#### Syntax

— ? — MM ————— %

#### Explanation

The ?MM command displays the status of the memory modules on the system.

#### Example

```
?MM  
MEMORY STATUS  
16 IN USE 0-15
```

## ?MSG

### Syntax

— ? <LSN>  
\* MSG \_\_\_\_\_ %

### Explanation

The ?MSG command displays the most recent messages associated with tasks that were run under the usercode of the station.

The syntax and results of this command are controlled to some extent by the CANDE *ALLMSG* option. The state of the *ALLMSG* option is controlled by the system operator.

If *ALLMSG* is RESET, a syntax error is issued if <LSN> or asterisk (\*) is specified. The only information returned pertains to those jobs or tasks originating from the user's LSN with which the usercode is associated.

If *ALLMSG* is SET, information about any LSN can be requested. The asterisk (\*) syntax is a shorthand method for specifying the LSN to which the user is signed on. If no LSN is specified, all information pertaining to jobs or tasks for the user making the request, regardless of origination point, is returned.

### Example

```
?MSG
----- MESSAGES -----
Ø123 OPERATOR DSED @ 5:ØØ1F:5*
Ø123 OPERATOR DISPLAY: BEGIN TEST ONE
```

# ?MXA

### Syntax

— ? — 

<LSN>
*

 MXA ————— %

### Explanation

The ?MXA command displays information about jobs running with the usercode of the station. The information may include the following:

- Scheduled, waiting, or error status
- Job number
- Mix number
- Priority
- Job name
- RSVP message
- Display message

The syntax and results of this command are controlled to some extent by the CANDE ALLMSG option. The state of the ALLMSG option is controlled by the system operator.

If ALLMSG is RESET, a syntax error is issued if <LSN> or asterisk (\*) is specified. The only information returned pertains to those jobs or tasks originating from the user's LSN with which the usercode is associated.

If ALLMSG is SET, information about any LSN can be requested. The asterisk (\*) syntax is a shorthand method for specifying the LSN to which the user is signed on. If no LSN is specified, all information pertaining to jobs or tasks for the user making the request, regardless of origination point, is returned.

### Example

```
?MXA
----- JOB STRUCTURE -----
*1516 JOB 99 COPY&COMPARE = FROM
W..1520 99 LIBRARY/MAINTENANCE
R: RECOPY REQD TEST/DATA
```



## ?NF

### Syntax

— ? —<mix number list>— NF —————%

### Explanation

The ?NF command can be used to respond to a “NO FILE” message. A “NO FILE” message is displayed when a file is sought whose OPTIONAL file attribute is set to FALSE. A ?NF reply returns an error result value of NOFILEFOUNDRSLT (2) to the user program and allows the program, if it is so designed, to handle the error condition without being terminated. If a mix number is not specified, the currently running program is assumed.

### Example

```
LIST
100 BEGIN
200 FILE      F(KIND=TAPE);
300 INTEGER I;
350
400 I := OPEN(F);
500 DISPLAY(String(I,*));
600 END.
#
```

```
RUN
#RUNNING 6392
#6392 NO FILE F (MT) #1
?NF
#6392 GOING
#6392 DISPLAY:2.
#ET=6.9 PT=0.3 IO=0.4
```

## **?NORESTART**

This command is documented under the ?RESTART command.

## ?NOTOK

### Syntax

— ? <mix number> NOTOK \_\_\_\_\_ %

### Explanation

The ?NOTOK command responds NO to the RSVP message of the suspended task, but allows the task to continue processing.

The ?NOTOK command can be used in response to certain suspended tasks. Use the ?Y command to inquire if the ?NOTOK response is applicable. If you do not specify a mix number, the currently running task is assumed.

### Example

```
LIST
2000 BEGIN
4000 FILE IN(KIND=DISK,DEPENDENTSPECS=TRUE),
5000   OUT(KIND=DISK,NEWFILE=TRUE);
6000 BOOLEAN PROCEDURE CMP(A,B);
8000   ARRAY A,B[0];
10000  BEGIN
12000  CMP := POINTER(A) LSS POINTER(B) FOR 80;
14000  END CMP;
15000 MYSELF.OPTION :=* & REAL(TRUE) [VALUE(SORTLIMITS):1];
16000 SORT(OUT,IN,0,CMP,15,90,0);
18000 LOCK(OUT,CRUNCH);
20000 END.
#
RUN
#RUNNING 2844
#2844 ENTER OK TO ALLOW SORT TO INCREASE MEMORY FROM 1410 TO 72090
?Y
STATUS OF TASK 2832/2844 AT 11:23:29
PRIORITY = 50
ORIGINATION = LSN 350
USERCODE = TEST
CHARGECODE = 6463
STACK STATE = WAITING ON AN EVENT
PROGRAM NAME = (TEST)OBJECT/SORT
RSVP = ENTER OK TO ALLOW SORT TO INCREASE MEMORY FROM 1410 TO 72090
REPLY = NOTOK,OK
?NOTOK
#2844 GOING
#2844 SORT ERROR # : 3 INSUFFICIENT MEMORY FOR MEMORY SORT
#P-DS @ 00016000.
#ET=19.0 PT=0.1 IO=0.1
```

## ?NUMBERED

### Syntax

— ? — NUMBERED  
UNNUMBERED ————— %

### Explanation

The ?NUMBERED and ?UNNUMBERED commands apply only to schedule sessions. If a ?NUMBERED command appears in the input file, the sequence number of each line in the input file, beginning with the first line following the command, appears at the extreme right margin of the image of that line in the output file (unless insufficient room exists on that line). The ?UNNUMBERED command restores the default option; namely, sequence numbers do not appear.

### Example

```
list file
#FILE (UZER)FILE ON USERPACK
2000 ?numbered
4000 ?report "numbered output"
#
```

```
sch
#UPDATING
#SCHEDULE #00021 RUNNING
OUTPUT FILE IS (UZER)SCHOUT/FILE ON USERPACK.
LINE WIDTH IS 72
CHARGECODE IS 1111.
#
```

```
list schout/file
#FILE (UZER)SCHOUT/FILE ON USERPACK
SESSION(S): 7321.
```

```
#B7900:277 CANDE 36.206 AT LF79A; YOU ARE SCHED#000(39)
```

```
#SESSION 7321 18:13:27 02/24/87
```

```
?numbered
```

```
?report "numbered output"
```

4000

```
#
```

```
#END SESSION 7321 ET=0.8 PT=0.0 IO=0.0
```

```
#USER = UZER CHARGE = 6821. 18:13:28 02/24/87
```

```
#
```

## ?OF

### Syntax

-- ? <mix number> OF \_\_\_\_\_%

### Explanation

The ?OF command can be used in response to a "#NO FILE" message where the file being sought has the attribute OPTIONAL set to TRUE. End-of-file action results from entering this command. If a <mix number> is not specified, the currently running program is assumed.

### Example

```
LIST
100 BEGIN
200 FILE F(KIND=DISK,OPTIONAL=TRUE);
300 F.OPEN:=TRUE;
400 END.
#

RUN
#RUNNING 1840
#1840 NO FILE(F) ON PACK(USERFILES)

?OF
#1840 GOING
#ET=7.2 PT=0.3 IO=0.4
```





## ?PURGE Control Command

---

# ?PURGE

### Syntax

— ? — PURGE ————— %

### Explanation

The ?PURGE command purges all queued input and sets the user's queued input state to normal. Any command in progress is not affected.

### Example

```
?DS
#0380 OPERATOR DSED @ 003:0002:5*
#O-DS @ 00000300.
#ET=1:14.4 PT=0.1 IO=0.1
#QUEUED INPUT PENDING

?PURGE
#QUEUED INPUT PURGED
```



## ?REPEAT

### Syntax

— ? — REPEAT <edit specs> %

### Explanation

The ?REPEAT command causes CANDE to process the user's saved text as if it were the next line of input. If editing specifications are provided, the text is edited and displayed before being processed. If queued input exists, it is processed when the repeated text finishes. This command cannot be used while the station is busy or in sequence mode, or in a file that is to be sent to a schedule station.

Refer to the FIX command in Section 3, "CANDE Commands," for an explanation of <edit specs>.

### Example

```
?SHOW ALL
[2]g don/1*
[1]run*
<0>list*
#

?REPEAT
#
100 BEGIN
200 STRING A;
300 ACCEPT(A);
400 END.
#
```

## ?REPORT Control Command

---

# ?REPORT

### Syntax

— ? — REPORT —<text>—————%

### Explanation

The ?REPORT command is valid only from schedule stations. A command of the form ?REPORT <text> sends the text as a message to any interactive CANDE stations that are logged on with the same usercode as the schedule session. This command acts as a restricted abbreviation of the ?TO <usercode> <text> command (where the <usercode> is understood). A number sign (#) acknowledgement appears in the schedule output file if the command is successful; a “#NOT ON” message appears if no interactive session with that usercode is underway.

### Example

```
M SALES/3
#WORKFILE SALES/3: SEQ
100?REPORT report printed

sch
#UPDATING
#SCHEDULE #00008 RUNNING
OUTPUT FILE IS (WIDGETS)SCHOUT/SALES/3 ON USERPACK
LINE WIDTH IS 72
#
#08:44 FROM SCHEDULE #00008: report printed
```

## ?RESTART

### Syntax

— ? —┐ RESTART ————— %  
          └─┘ NORESTART ┘

### Explanation

If a schedule session is aborted because of system failure, the session is restarted after the last ?RESTART command that was encountered prior to the failure. If no ?RESTART command appeared or if none appeared since a ?NORESTART command, the session is not restarted. If a schedule session is aborted without restart and an unsaved work file existed, a normal CANDE recovery file is generated. That work file can be recovered with a RECOVER command in either an interactive or schedule session.

### Examples

?REST

?NORE

# ?RESUME

### Syntax

— ? — RESUME \_\_\_\_\_%

### Explanation

The ?RESUME command is valid only from a schedule station. Any error detected in CANDE syntax, editing, compilation, or abnormal termination of execution causes the remainder of the input lines to be skipped until a ?RESUME command is found. If no ?RESUME command is found, the session is aborted. If an unsaved work file exists at that time, a recovery file is generated and can be recovered in an interactive or schedule session. (An explicit or default BYE, found with an unsaved work file, causes such error action.)

### Example

```
list test
#FILE (UZER)TEST ON USERPACK
100 COMPILE TESTIT ALGOL; ALGOL FILE CARD(KIND=DISK,TITLE=DON)
200 ?REPORT "LINE AFTER COMPILE"
300 ?RESUME
400 ?REPORT "LINE AFTER RESUME"
#
```

```
list don
#FILE (UZER)DON ON USERPACK
100 BEGIN
200 INTEGER I;
300 J:=1;
400 END.
#
```

```
sch
#SCHEDULE #00014 RUNNING
OUTPUT FILE IS (UZER)SCHOUT/TEST ON USERPACK
LINE WIDTH IS 72
#
#09:31 FROM SCHEDULE #00014: "LINE AFTER RESUME"
```

## ?RETRIEVE

### Syntax

— ? — RETRIEVE <ste#> REPEAT <edit specs> %

### Explanation

The ?RETRIEVE command retrieves the saved text entry indicated by the saved text entry number <ste#> from its current position in the user's saved text queue and makes it the most recent entry. The specified <ste#> must be greater than 0 and less than or equal to the greatest <ste#>. If no <ste#> is provided, the number 1 is assumed.

If ?REPEAT is specified, the retrieved saved text is processed as the next line of input. If editing specifications are provided, this text is edited and displayed before being processed. Refer to the FIX command in Section 3, "CANDE Commands," for an explanation of <edit specs>.

The ?RETRIEVE command cannot be used while the station is busy or is in sequence mode.

### Example

```
?SHOW ALL
[2]rem sch/payroll/3*
[1]run*
<0>list*
#
?RETRIEVE 2
#
?SHOW ALL
[2]run*
[1]list*
<0>rem sch/payroll/3*
#
```

## ?RM Control Command

---

### ?RM

#### Syntax

— ? <mix number> RM \_\_\_\_\_ %

#### Explanation

The ?RM command is used in response to a “#DUP LIBRARY” message. The old file is removed and the new file is retained. If a <mix number> is not specified, the currently running program is assumed.

#### Example

```
COPY & COMPARE TEST AS TEST/1 FROM USERPACK(PACK) TO USERPACK(PACK)
#RUNNING 1187
#BOT 1188 (UZER)WFLCODE
#BOT 1189 LIBRARY/MAINTENANCE
#1189 (UZER)TEST/1 DUP LIBRARY ON USERPACK PK068 *

?1189 RM
#1189 (UZER)TEST/1 REMOVED ON USERPACK PK068 .
#1189 (UZER)TEST/1 COPIED.
#EOT 1189 LIBRARY/MAINTENANCE
#EOT 1188 (UZER)WFLCODE
#
```

## **?RO**

The ?RO (reset options) command is described with the ?SO (set options) command.

## ?S Control Command

---

### ?S

#### Syntax

— ? <LSN>  
\* S \_\_\_\_\_ %

#### Explanation

The ?S command lists any scheduled jobs and tasks in the mix.

The syntax and results of this command are controlled to some extent by the CANDE *ALLMSG* option. The state of the *ALLMSG* option is controlled by the system operator.

If *ALLMSG* is *RESET*, a syntax error is issued if an LSN or an asterisk (\*) is specified. The only information returned pertains only to those jobs or tasks originating from the LSN associated with the requestor's usercode.

If *ALLMSG* is *SET*, information about any LSN may be requested. The \* syntax is a shorthand method for specifying the LSN to which the user is signed on. If no LSN is supplied, all information pertaining to jobs or tasks for the user making the request, regardless of origination point, is returned.

#### Example

```
?S
---Mix-Pri---Elapsed----- 1 SCHEDULED ENTRY ( ALL )USER=SMITH
* 7204 50      :03 (SMITH)OBJECT/UTIL ON PACK1
```



## ?SC

### Syntax

— ? — SC \_\_\_\_\_%

### Explanation

The ?SC command displays the current system configuration.

### Example

```
?SC
2 PROCESSORS 1-2
DØ=Ø21ØØ
2 MULTIPLEXORS 1-2
MULTIPLEXOR LOAD LIMIT
  MPX          LIMIT  TRAFFIC
  1 (MOD II)   15     N/A
  2 (MOD II)   15     N/A
MEMORY STATUS
  16 IN USE Ø-15
PROGRAMMED HALT/LOADS: Ø
STRINGS PRESENT: 1-3
```



## ?SCHEDULE

### Syntax

— ? — SCHEDULE \_\_\_\_\_%

### Explanation

The ?SCHEDULE command lists each schedule session currently active or scheduled with the same usercode as that of the station where the command is entered. Whether or not any such sessions exist, a summary of the schedule facility status is produced.

Each schedule session is identified by its schedule number, and an indication of its status is given. The status regarding a schedule session is explained in the following table:

Status	Meaning
SN=1234	Active session, logged-on as session #1234
RUNNING	Active session, logging-on or off
WAITING	Session waiting for schedule station
18:30	Session to be run after 6:30 pm
06:30T	Session to be run after 6:30 am tomorrow
(12:00)	Session to have been run after noon, now waiting
RESTART	Session awaiting restart

The summary of the schedule facility status displays the following:

- The current limit (number of schedule sessions CANDE can run at once)
- The number of schedule stations attached to CANDE (if different from the limit)
- The number of active sessions
- The number of sessions ready (waiting for a station)
- The number pending (waiting for an *after* time)
- The time for the first pending station

Except for the limit, no numbers are suppressed. The schedule facility is not initialized automatically until one minute after CANDE is started or restarted. In this case, the summary says “#SCHEDULE NOT YET INITIALIZED; LIMIT = n” (where *n* is the limit CANDE attempts to establish when initialization takes place).

### Example

```
?SCHE
#00020 SN=1123
#SCHEDULE LIMIT=5 NO USERLIMIT ACTIVE=1
```

## **?SF**

### **Syntax**

— ? — SF ————— %

### **Explanation**

The ?SF command displays the value of the memory management parameters.

### **Example**

```
?SF
1) OLAY GOAL = 1% PER MINUTE
2) AVAILMIN = 0%
3) FACTOR = 150%
4) MEM PRIORITY FACTOR = 2%
```

## ?SHOW

### Syntax

```
— ? — SHOW _____%  
      |  
      | QUEUED  
      | SAVED  
      | ALL
```

### Explanation

The ?SHOW command causes the saved text queue and/or the visible portion of the input queue to be displayed. Each line displayed is followed by an asterisk (\*) to indicate the last character of text. If no modifiers are supplied, the user's saved text is displayed. If the QUEUED option is specified, only the user's visible queued input is displayed. The SAVED option displays the user's entire saved text queue, and the ALL option displays both the saved text queue and the visible queued input.

When the ALL, QUEUED, and SAVED options are specified, each element displayed is preceded by its relative number. The saved text queue entries that are not the most recent are preceded by the queue entry number enclosed in square brackets ([ ]). The most recent saved text entry is preceded by <0>, and each queued entry is preceded by its queue entry number in parentheses.

A W preceding a queued input entry indicates that CANDE is waiting for a user response before taking the queued input entry out of the queue and performing it. (Refer to the ?ENTER queue manipulation command in this section.)

The ?SHOW command cannot be used while the station is busy or is in sequence mode.

### Examples

```
?SHOW  
sch sch/patch*
```

```
?SHOW ALL  
[4]list inventory/1*  
[3]g inventory/1*  
[2]?ds*  
[1]list*  
<0>m inventory/2*  
#
```

## ?SI Control Command

---

### ?SI

#### Syntax

— ? — SI ————— %

#### Explanation

The ?SI command displays the name of the intrinsics file currently in use by the system.

#### Example

```
?SI  
INTRINSICS: SYSTEM/INTRINSICS ON PACK
```

## ?SL

### Syntax

— ? — SL \_\_\_\_\_ %

### Explanation

The ?SL command lists the support library functions currently available to the system.

### Example

```
?SL
SL HELPSUPPORT = *SYSTEM/HELP ON DISK
SL RPGSUPPORT  = *SYSTEM/RPGSUPPORT
SL PLISUPPORT  = *SYSTEM/PLISUPPORT
SL XREFSUPPORT = *SYSTEM/XREFSUPPORT ON USERMASK
```

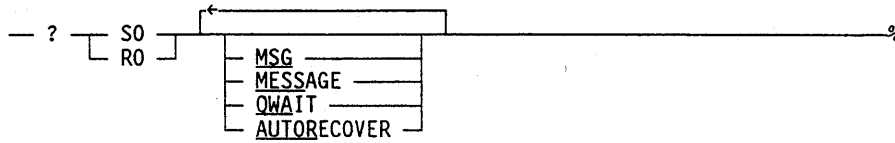
Support library functions are established via the ODT *SL* command. Refer to the *A Series System Commands Operations Reference Manual* for additional information.

## ?SO Control Command

---

### ?SO or ?RO

#### Syntax



#### Explanation

The ?SO and ?RO commands have the same explanations as the CANDE commands SO and RO, except that the ?SO and ?RO commands are available for use when a station is busy.



# ?SQ

## Syntax

— ? <LSN>  
\* SQ <queue number> — %

<queue number>

— <integer> —

## Explanation

The ?SQ command displays jobs queued to run that were initiated under the usercode of the station. If a <queue number> is not specified, entries in all job queues are displayed. Otherwise, only entries in the queue referenced by the <queue number> are displayed.

If a job is specified after a BEGIN JOB statement, the name of the job waiting in the queue is displayed. If a job has parameters, the job name is displayed first, followed by the actual parameters. If no job name is specified, then "BEGIN JOB;" is displayed, followed by the first 30 characters from the next line, which creates the first job attribute, declaration, or statement of the job.

The syntax and results of this command are controlled to some extent by the CANDE ALLMSG option. The state of the ALLMSG option is controlled by the system operator.

If ALLMSG is RESET, a syntax error is issued if <LSN> or an asterisk (\*) is specified. The only information returned pertains to those jobs or tasks originating from the user's LSN with which the usercode is associated.

If ALLMSG is SET, information about any LSN can be requested. The asterisk (\*) syntax is a shorthand method for specifying the LSN to which the user is signed on. If no LSN is specified, then all information pertaining to jobs or tasks for the user making the request, regardless of origination point, is returned.

## Example

```
?SQ Ø  
QUEUE Ø:  
NO ENTRIES
```

## ?SS Control Command

---

### ?SS

#### Syntax

```
-- ? -- SS --> <LSN> -----> <text> -----> %
                |
                |-----> <NLS>
                |
                |-----> <station name>
                |
                |-----> SPO
```

#### Explanation

The ?SS command sends a message to another CANDE station. If an LSN of 1 (which is the ODT) or SPO is specified, the message is sent to the ODT.

If a message is to be sent to a log station, it must be addressed to the LSN of that log station. Log stations can receive messages sent to the SPO when LGSPO (an option of the CANDE LGSTA command) is set. Refer to the LGSTA command in the *A Series CANDE Configuration Reference Manual* for additional information. Messages sent from a log station are identified "FROM <user> ON <LSN>."

#### Examples

```
?SS 1 please mount tape number 92745
#
```

```
?SS SPO WHAT IS THE SCHEDULE FOR CANDE SERVICE?
#
```

```
?SS 63 Your program does not work
#
```

## ?ST

### Syntax

— ? <mix number> ST ————— %

### Explanation

The ?ST command suspends the referenced task. If a <mix number> is not specified, the currently running program is assumed. A suspended task can be resumed with the ?OK command.

### Example

```
list
100 100 I=1
200 GO TO 100
300 STOP
400 END

run
#RUNNING 1155

?st
#1155 OPERATOR STOPPED

?OK
#1155 GOING
```

## ?STARTTIME Control Command

---

### ?STARTTIME

#### Syntax

```
-- ? --<mix number list> STARTTIME -- = --<time> -- ON --<date> %
```

#### <date>

```
--<mm> / --<dd> / --<yy>
```

#### <mm>

```
--/2\--<digit>
```

#### <dd>

```
--/2\--<digit>
```

#### <yy>

```
--/2*\--<digit>
```

#### Explanation

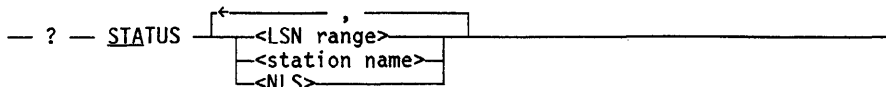
The ?STARTTIME command assigns a start time to a queued job. This command can be used whether or not the job already has a start time. If the job reaches the front of the queue and the current time is not greater than or equal to the specified start time, the job retains its position at the head of the queue. Refer to the START command for a definition of <time>. The CONTROLLER selects the first job in the queue whose start time is less than or equal to the current time as the next job to be run from that queue. For further information on the handling of jobs in queues, refer to the STARTTIME command in the *System Commands Reference Manual*.

#### Example

```
?6152 STARTTIME = 0930
```

# ?STATUS

## Syntax



## Explanation

The ?STATUS command displays the status of either the currently running task or any station on the data comm network. If ?STATUS is entered without any other specifications, CANDE displays one of the following:

- The status of the current program
- The status of the CANDE command (for example, LIST, FIND, and REPLACE)
- The position of the program or command in the work, task, or resource queue

If the ?STATUS command is used to display the status of any station on the data comm network, CANDE returns information about the ENABLED, ATTACHED, READY, and/or SWITCHED states of the designated stations. The LSN range, station name, and NLS of the station are also listed. If the station is not owned by CANDE, the name of the controlling MCS is given in square brackets. For pseudostations and schedule stations, the NLS number is replaced by PSEUDO\_STA or SCHEDULE\_STA, and the READY and ENABLED states are not shown.

*Note: The minimum abbreviation for ?STATUS is ?STA. ?ST is a separate command that will suspend a task. (Refer to ?ST for more information.)*

## Examples

```
?STA
#12:21 ET=19:12:04 PT=0:12:03 IO=0:0:11
```

```
?STA 95
SHERRI(95)=0:65:0 RDY ENAB ATT
```

```
?STA
#15:55 3RD IN RESOURCE QUEUE
```

**?STATUS Control Command (cont.)**

---

?STA 2,5,9-12,0:10,0:14,TD8441,TTY1  
TTY0(2)NOLINE RDY ENAB UNATT  
TTY3(5)=0:7:0 RDY ENAB ATT SW=DISCON  
TD8441(9)=0:3:1 RDY ENAB ATT  
TD0013(10)=0:4:1 RDY ENAB ATT  
TD9333(11)=0:9:6 RDY ENAB ATT  
TD8450(12)=0:3:3 RDY ENAB ATT  
TD81200(60)=0:10:0 RDY ENAB ATT SW=DISCON  
TTY1200(61)=0:14:0 UNRDY ENAB ATT  
TD8441(9)=0:3:1 RDY ENAB ATT  
TTY1(3)=0:5:0 RDY ENAB ATT SW=CONN

## ?STUP

### Syntax

— ? — STUP —————\*

### Explanation

The ?STUP command displays the current state of the startup file facility. If the facility is in use, the current < startup name > is also shown.

For additional information, refer to “Startup Files and Restart Files” in Section 1, “General Information,” and to the ?STUP command in the *A Series CANDE Configuration Reference Manual*.

### Examples

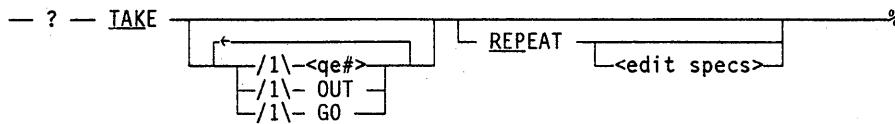
```
?STUP
# STARTUP NAME: CANDE/STARTUP; RESTART NAME: CANDE/STARTUP/RESTART
```

## ?TAKE Control Command

---

### ?TAKE

#### Syntax



<qe#>

<integer>

#### Explanation

The ?TAKE command takes an entry from the visible portion of the user's queue. If the OUT option is specified, the entry is thrown away; otherwise, it is entered in the user's saved text queue. If no queue entry number (<qe#>) is supplied, the first entry in the queue is taken.

The GO option resumes the session with the first entry in the queue. The GO option must not be followed by the REPEAT option.

The REPEAT option resumes the session with the saved text. If editing specifications are provided, the saved text is edited and displayed before being processed. Refer to the FIX command in Section 3, "CANDE Commands," for an explanation of <edit specs>.

If no entries remain in the queue after completion of the ?TAKE command, the queued input state is set to normal.

The ?TAKE command cannot be used while the station is busy or is in single-line sequencing or page mode.



**Example**

```
?SHOW  
fix 120.A.BCD*  
#
```

```
?SHOW QUEUED  
(1)REM*  
(2)make a a*  
w(3)1 A/B ON PACK*  
(4)100 TEXT FOR A*  
#
```

```
?take 2  
#
```

```
?sh  
make a a*  
#
```

## ?TD Control Command

---

### ?TD

#### Syntax

— ? — TD \_\_\_\_\_%

#### Explanation

The ?TD command displays the current date, time and time zone.

#### Example

?TD

The date is Tuesday, March 19, 1991

The time is 16:11:43 Pacific Standard Time ( PST )

## ?TF

### Syntax

— ? — TF \_\_\_\_\_%

### Explanation

The ?TF command displays the value of the memory management parameters.

*Note: The ?TF command will be deimplemented in a future release. The ?SF command replaces the function of the ?TF command. See the ?SF command for more information in this section.*

### Example

```
?TF
1) OLAY GOAL = 1% PER MINUTE
2) AVAILMIN = 0%
3) FACTOR = 150%
4) MEM PRIORITY FACTOR = 2%
```

## ?TI Control Command

---

### ?TI

#### Syntax

— ? <mix number> TI — %

#### Explanation

The ?TI command displays the process time, elapsed time, and I/O time expended by the referenced task. The time and number of operations allocating initial off-stack items (such as arrays and strings) are also included. If a <mix number> is not specified, the currently running program is assumed.

This information can also be seen with the ?STATUS command.

#### Example

```
?TI
TIMES FOR 4853
PROCESS  =0:00:01
IO       =0:00:02
READYQ   =0:00:01
INITPBIT =0:00:00 170 OPERATIONS
OTHERBIT =0:00:01
ELAPSED  =0:00:01
```

## ?TIME

### Syntax

— ? — TIME \_\_\_\_\_%

### Explanation

The ?TIME command displays the current time and date. Time zone information is also displayed if a time zone is configured for the system. The system command TR (Time Reset) can be used to set the time zone for the system. Refer to the *A Series System Commands Operations Reference Manual* for more information about configuring time zones.

### Example

```
?time  
#10:25 AM TUESDAY, MARCH 29, 1977
```

```
?time  
#4:20:17 PM WEDNESDAY, FEBRUARY 20, 1991 Pacific Standard Time (PST)  
[12:20 AM THURSDAY, FEBRUARY 21, 1991 UNIVERSAL TIME]
```

## ?TO Control Command

---

# ?TO

### Syntax

— ? — TO —<usercode>—<text>—————%

### Explanation

The ?TO command sends the specified <text> to all interactive stations where <usercode> is logged on.

If <usercode> is not logged on at any interactive station, CANDE responds with “# <usercode> NOT ON.” Otherwise, CANDE responds with a number sign (#).

### Example

?to purchasing your log sheets are due tomorrow.  
#

A message similar to the following will appear at the station or stations at which the usercode PURCHASING is logged on:

#15:40 FROM ACCOUNTING ON 11: your log sheets are due tomorrow.  
#

## **?UNNUMBERED**

This command is documented under the ?NUMBERED command.

## ?W Control Command

---

### ?W

#### Syntax

— ? <LSN>  
\* W \_\_\_\_\_ %

#### Explanation

The ?W command lists any waiting jobs and tasks in the mix.

The syntax and results of this command are controlled to some extent by the CANDE *ALLMSG* option. The state of the *ALLMSG* option is controlled by the system operator.

If *ALLMSG* is *RESET*, a syntax error is issued if an LSN or an asterisk (\*) is specified. The only information returned pertains only to those jobs or tasks originating from the LSN associated with the requestor's usercode.

If *ALLMSG* is *SET*, information about any LSN may be requested. The \* syntax is a shorthand method for specifying the LSN to which the user is signed on. If no LSN is supplied, all information pertaining to jobs or tasks for the user making the request, regardless of origination point, is returned.

#### Example

```
?W
---Job-Task-Pri---Elapsed----- 1 WAITING ENTRY USER=SMITH -----
* 1448\1537 50      :08 (SMITH) (SMITH)OBJECT/UTIL ON PACK1
      PROGRAMATICALLY SUSPENDED
```



## ?WAIT

### Syntax

— ? — WAIT \_\_\_\_\_%

### Explanation

The ?WAIT command sets the state of the user's queued input to waiting. Any command in progress is not affected.

### Example

```
E
#RUNNING 5328

SA
#QUEUED

?WAIT
#
#ET=11.2 PT=0.1 IO=0.6
#QUEUED INPUT WAITING

WHAT
#QUEUED, QUEUED INPUT WAITING

?go
#
#WORKFILES ALREADY SAVED
#WORKFILES TEST: ALGOL, 13 RECORDS (THRU 900), SAVED
#OBJECT FILE PRESENT, SAVED
```

**?WAIT Control Command (cont.)**

---

## ?WD

### Syntax

— ? — WD ————— %

### Explanation

The ?WD command displays the current date.

*Note:* The ?WD command will be deimplemented on a future release. Use the ?TD command to display the current date and time of day.

### Example

The date is Tuesday, March 19, 1991

## ?WHERE Control Command

---

# ?WHERE

### Syntax

— ? — WHERE —<usercode>—————%

### Explanation

The ?WHERE command displays the station name and logical station number of all stations at which <usercode> is logged on.

### Examples

```
?WH JOHNSON  
# JOHNSON ON SHERRI(95)  
# JOHNSON ON TTYØ(15)
```

```
?WH ALAN  
# ALAN NOT ON
```

## ?WHY and ?Y

### Syntax

-- ? [ <mix number> ] [ WHY ] [ Y ] %

### Explanation

The ?WHY command displays pertinent information about the referenced task. If a <mix number> is not specified, the currently running task is assumed. ?Y is a synonym for ?WHY.

### Example

```
?WHY
STATUS OF TASK 4226 AT 11:34:15
PRIORITY = 50
ORIGINATION: <LSN> 96
STACK STATE: HOLDING
SWAP STATUS: ON DISK
PROGRAM: OBJECT/ED ON TIOADMIN
```

## ?WM Control Command

---

### ?WM

#### Syntax

— ? — WM \_\_\_\_\_ %

#### Explanation

The ?WM command displays information about the master control program (MCP) currently running on the system.

#### Example

```
MCP: *SYSTEM/MCP/36194/DIAG 36.194.4361
H/L UNIT: 47
COMPILED: 12/03/86 @ 05:02:06 (NEWP 36.194)
  COMPILETIME OPTIONS ARE:
    TRACE           DIAGNOSTICS       EXPERIMENTAL
    LINEINFO        LOCKTRACE         READLOCK
    READLOCKTIMEOUT RESTART           SWAPTRACE
H/L REASON: MANUAL
H/L TIME: WEDNESDAY DEC 10,1986 (86344) 7:53 PM.
GROUP ID: LF79A
HOSTNAME: LF79A
SYSTEM SERIAL NO: 178
CATALOG LEVEL: 0
NEXT MCP: NOT SPECIFIED
CONTROLWARE TYPES SAVED: B98387 B9389 B9387S 1 AM.
#
```

## ?WRU

### Syntax

— ? — WRU ————— %

### Explanation

The ?WRU command displays identification of the station as well as the version of CANDE that is running. If a user is logged on at the station, the session number and usercode are displayed. If a user is logged on and has a chargecode, the chargecode is also displayed.

### Example

```
?WRU
#A15:277 CANDE 37.200 AT HOSTESS; YOU ARE STA47(95)
#SESSION = 0942 USER = JOHNSON
```

## ?WS Control Command

---

### ?WS

#### Syntax

— ? — WS \_\_\_\_\_%

#### Explanation

The ?WS message displays the name of the current system supervisor program.

*Note: The ?WS command will be deimplemented on a future release. The ?CS command replaces the function of the ?WS command. See the ?CS command for more information in this section.*

#### Example

```
?WS  
SUPERVISOR: SYSTEM/SUP ON CONTROL
```



## ?WT

### Syntax

— ? — WT \_\_\_\_\_ %

### Explanation

The ?WT command displays the current time of day and time zone.

*Note: The ?WT command will be deimplemented on a future release. Use the ?TD command to display the current date and time of day.*

### Example

?WT  
The time is 15:59:00 Pacific Standard Time ( PST )

## **?Y Control Command**

---

### **?Y**

?Y and ?WHY are synonyms. (Refer to the ?WHY command.)

# Section 5

## Functional Command Groupings

This section provides groupings of CANDE commands for functional reference purposes.

### Task Commands

ADD BIND COMPILE COPY EXECUTE RUN START UTILITY WFL

### Library Commands

ALTER CHANGE FILES LFILES REMOVE SECURITY TITLE TYPE

### Work File Commands

DISCARD GET LOAD MAKE RECOVER SAVE UPDATE WHAT

### Search Commands

FIND LIST MATCH RANGE

### Edit Commands

DELETE EXCLUDE FIX INSERT MARKID MERGE MOVE RENEW  
REPLACE RESEQ RMERGE SEQ Single-line-entry TAB TAPE

### Page Mode Commands

NEXT PAGE SEQ SAME VOID + -

### User Information

ACCESS APASSWORD CHARGE CONVENTION CONTINUE DESTNAME  
ESCAPE FAMILY LANGUAGE MARGIN PASSWORD RO SO ?RO ?SO

### Utility Commands

BACKUPPROCESS DCSTATUS LOG NEWS WRITE

### Session Commands

BYE HELLO SCHEDULE SPLIT STATUS STOP

## Functional Command Groupings

---

### System Inquiry Commands

?COUNTS ?ID ?MM ?MSG ?SC ?SCHEDULE ?SI ?SL ?STARTTIME  
?STATUS ?STUP ?TD ?TF ?TIME ?WM ?WS ?WT

### Task Interrogation and Control Commands

?ASDU ?AX ?BREAKPOINT ?C ?CS ?CU ?DS ?DUMP  
?FA ?FR ?HI ?JA ?LIBS ?MXA ?NF ?NOTOK ?OF  
?OK ?OT ?RM ?S ?SQ ?ST ?TI ?W ?WHY ?Y

### Schedule Control Commands

?EOL ?NORESTART ?NUMBERED ?REPORT ?RESTART  
?RESUME ?UNNUMBERED ?% (COMMENT)

### End-of-File Commands

?DENY ?END

### Data Comm Network Commands

?AT CONNECT ?CONNECT ?HN MCS ?MCS ?SS ?TO ?WHERE  
?WRU

### Input Editing

?EDIT ?REPEAT ?RETRIEVE ?SHOW

### Queue Managing

DO ?ENTER ?GO ?PURGE ?TAKE ?WAIT

### Terminal Information

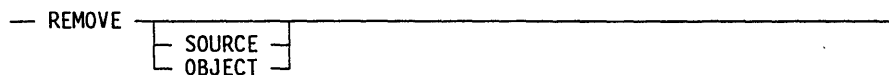
?CHAR ECHO TERMINAL

# Appendix A

## Understanding Railroad Diagrams

### What Are Railroad Diagrams?

Railroad diagrams are diagrams that show you the rules for putting words and symbols together into commands and statements that the computer can understand. These diagrams consist of a series of paths that show the allowable structure, constants, and variables for a command or a statement. Paths show the order in which the command or statement is constructed. Paths are represented by horizontal and vertical lines. Many railroad diagrams have a number of different paths you can take to get to the end of the diagram. For example:



If you follow this railroad diagram from left to right, you will discover three acceptable commands. These commands are

- REMOVE
- REMOVE SOURCE
- REMOVE OBJECT

If all railroad diagrams were this simple, this explanation could end here. However, because the allowed ways of communicating with the computer can be complex, railroad diagrams sometimes must also be complex.

Regardless of the level of complexity, all railroad diagrams are visual representations of commands and statements. Railroad diagrams are intended to

- Show the mandatory items.
- Show the user-selected items.
- Present the order in which the items must appear.
- Show the number of times an item can be repeated.
- Show the necessary punctuation.

To familiarize you with railroad diagrams, this explanation describes the elements of the diagrams and provides examples.

Some of the actual railroad diagrams you will encounter might be more complex. However, all railroad diagrams, simple or complex, follow the same basic rules. They

## Understanding Railroad Diagrams

---

all consist of paths that represent the allowable structure, constants, and variables for commands and statements.

By following railroad diagrams, you can easily understand the correct syntax for commands and statements. Once you become proficient in the use of railroad notation, the diagrams serve as quick references to the commands and statements.

### Constants and Variables

A constant is an item that cannot be altered. You must enter the constant as it appears in the diagram, either in full or as an allowable abbreviation. If a constant is partially underlined, you can abbreviate the constant by entering only the underlined letters. In addition to the underlined letters, any of the remaining letters can be entered. If no part of the constant is underlined, the constant cannot be abbreviated. Constants can be recognized by the fact that they are never enclosed in angle brackets (< >) and are in uppercase letters.

A variable is an item that represents data. You can replace the variable with data that meets the requirements of the particular command or statement. When replacing a variable with data, you must follow the rules defined for the particular command or statement. Variables appear in railroad diagrams enclosed in angle brackets.

In the following example, BEGIN and END are constants while <statement list> is a variable. The constant BEGIN can be abbreviated since it is partially underlined. Valid abbreviations for BEGIN are BE, BEG, and BEGI.

— BEGIN —<statement list>— END —————|

### Constraints

Constraints are used in a railroad diagram to control progression through the diagram. Constraints consist of symbols and unique railroad diagram line paths. They include

- Vertical bars
- Percent signs
- Right arrows
- Required items
- User-selected items
- Loops
- Bridges

A description of each item follows.

#### Vertical Bar

The vertical bar symbol (|) represents the end of a railroad diagram and indicates the command or statement can be followed by another command or statement.

— SECONDWORD — ( —<arithmetic expression>— ) —————|

### Percent Sign

The percent sign (%) represents the end of a railroad diagram and indicates the command or statement must be on a line by itself.

— STOP —————|%

### Right Arrow

The right arrow symbol (>) is used when the railroad diagram is too long to fit on one line and must continue on the next. A right arrow appears at the end of the first line, and another right arrow appears at the beginning of the next line.

— SCALERIGHT — ( —<arithmetic expression>— , —————>  
 >—<arithmetic expression>— ) —————|

### Required Items

A required item can be either a constant, a variable, or punctuation. A required item appears as a single entry, by itself or with other items, on a horizontal line. Required items can also exist on horizontal lines within alternate paths or nested (lower-level) diagrams. If the path you are following contains a required item, you must enter the item in the command or statement; the required item cannot be omitted.

In the following example, the word EVENT is a required constant and <identifier> is a required variable:

— EVENT —<identifier>—————|

### User-Selected Items

User-selected items appear one below the other in a vertical list. You can choose any one of the items from the list. If the list also contains an empty path (solid line), none of the choices are required. A user-selected item can be either a constant, a variable, or punctuation. In the following railroad diagram, either the plus sign (+) or the minus sign (-) can be entered before the required variable <arithmetic expression>, or the symbols can be disregarded because the diagram also contains an empty path.

+
-

 <arithmetic expression>—————|

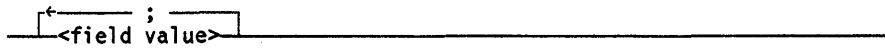
# Understanding Railroad Diagrams

---

## Loop

A loop represents an item or group of items that you can repeat. A loop can span all or part of a railroad diagram. It always consists of at least two horizontal lines, one below the other, connected on both sides by vertical lines. The top line is a right-to-left path that contains information about repeating the loop.

Some loops include a return character. A return character is a character – often a comma (,) or semicolon (;) – required before each repetition of a loop. If there is no return character, the items must be separated by one or more blank spaces.

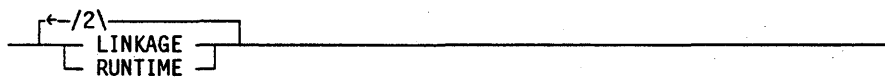


## Bridge

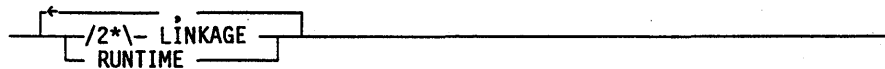
Sometimes a loop also includes a bridge, which is used to show the maximum number of times the loop can be repeated. The bridge can precede the contents of the loop, or it can precede the return character (if any) on the upper line of the loop.

The bridge determines the number of times you can cross that point in the diagram. The bridge is an integer enclosed in sloping lines (/ \). Not all loops have bridges. Those that do not can be repeated any number of times until all valid entries have been used.

In the first bridge example, you can enter LINKAGE or RUNTIME no more than two times. In the second bridge example, you can enter LINKAGE or RUNTIME no more than three times.



In some bridges an asterisk (\*) follows the number. The asterisk means that you must cross that point in the diagram at least once. The maximum number of times that you can cross that point is indicated by the number in the bridge.



In the previous bridge example, you must enter LINKAGE at least once but no more than twice, and you can enter RUNTIME any number of times.

The following figure shows the types of constraints used in railroad diagrams.



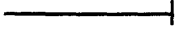
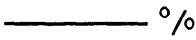


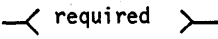
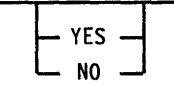
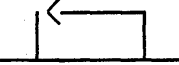
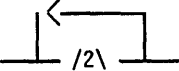
SYMBOL/PATH	EXPLANATION
	Vertical bar. Indicates that the command or statement can be followed by another command or statement.
	Percent sign. Indicates that the command or statement must be on a line by itself.
 	Right arrow. Indicates that the diagram occupies more than one line.
	Required items. Indicates the constants, variables, and punctuation that must be entered in a command or statement.
	User-selected items. Indicates the items that appear one below the other in a vertical list. You select which item or items to include.
	A loop. Indicates an item or group of items that can be repeated.
	A bridge. Indicates the maximum number of times a loop can be repeated.

Figure A-1. Railroad Constraints

## Following the Paths of a Railroad Diagram

The paths of a railroad diagram lead you through the command or statement from beginning to end. Some railroad diagrams have only one path, while others have several alternate paths. The following railroad diagram indicates there is only one path that requires the constant LINKAGE and the variable <linkage mnemonic>:

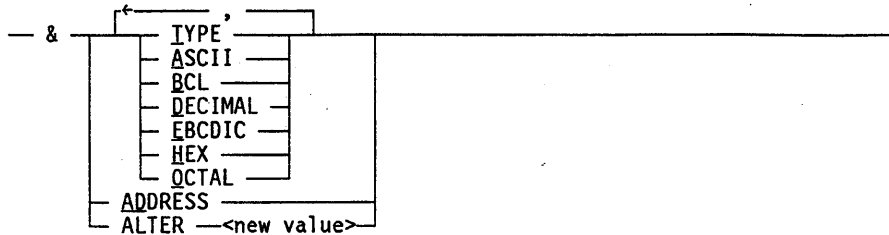
— LINKAGE —<linkage mnemonic>—————|

Alternate paths provide choices in the construction of commands and statements. Alternate paths are provided by loops, user-selected items, or a combination of both. More complex railroad diagrams can consist of many alternate paths, or nested (lower-level) diagrams, that show a further level of detail.

For example, the following railroad diagram consists of a top path and two alternate paths. The top path includes an ampersand (&) and the constants (that are

## Understanding Railroad Diagrams

user-selected items) in the vertical list. These constants are within a loop that can be repeated any number of times until all options have been selected. The first alternate path requires the ampersand and the required constant ADDRESS. The second alternate path requires the ampersand followed by the required constant ALTER and the required variable <new value>.



## Railroad Diagram Examples with Sample Input

The following examples show five railroad diagrams and possible command and statement constructions based on the paths of these diagrams.

### Example 1

<lock statement>



#### Sample Input

LOCK (FILE4)

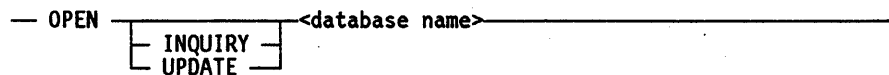
#### Explanation

LOCK is a constant and cannot be altered. Because no part of the word is underlined, the entire word must be entered.

The parentheses are required punctuation, and FILE4 is a sample file identifier.

### Example 2

<open statement>



#### Sample Input

OPEN DATABASE1

#### Explanation

The constant OPEN is followed by the variable DATABASE1, which is a database name.

The railroad diagram shows two user-selected items, INQUIRY and UPDATE. However, because there is an empty path (solid line), these entries are not required.

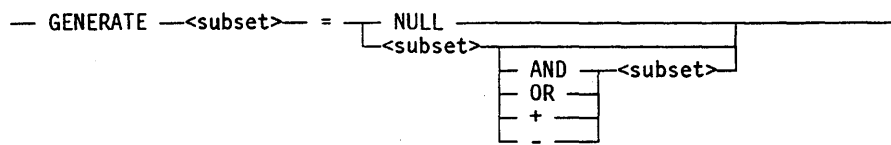
*continued*

*continued*

Sample Input	Explanation
OPEN INQUIRY DATABASE1	The constant OPEN is followed by the user-selected constant INQUIRY and the variable DATABASE1.
OPEN UPDATE DATABASE1	The constant OPEN is followed by the user-selected constant UPDATE and the variable DATABASE1.

### Example 3

**<generate statement>**

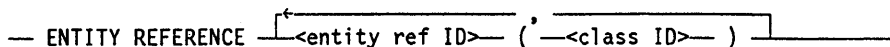


Sample Input	Explanation
GENERATE Z = NULL	The GENERATE constant is followed by the variable Z, an equal sign (=), and the user-selected constant NULL.
GENERATE Z = X	The GENERATE constant is followed by the variable Z, an equal sign, and the user-selected variable X.
GENERATE Z = X AND B	The GENERATE constant is followed by the variable Z, an equal sign, the user-selected variable X, the AND command (from the list of user-selected items in the nested path), and a third variable, B.
GENERATE Z = X + B	The GENERATE constant is followed by the variable Z, an equal sign, the user-selected variable X, the plus sign (from the list of user-selected items in the nested path), and a third variable, B.

# Understanding Railroad Diagrams

## Example 4

<entity reference declaration>



### Sample Input

ENTITY REFERENCE ADVISOR1 (INSTRUCTOR)

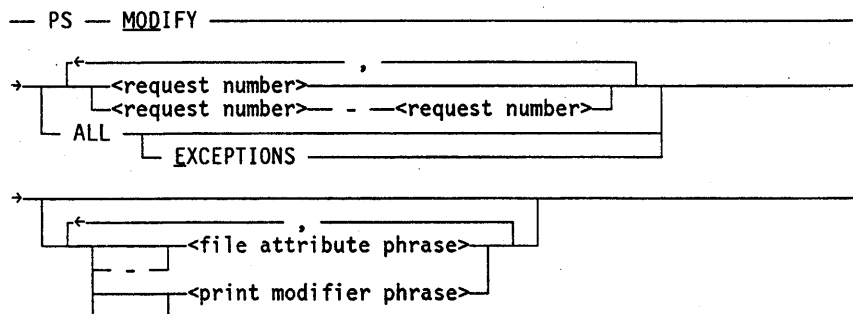
### Explanation

The required item ENTITY REFERENCE is followed by the variable ADVISOR1 and the variable INSTRUCTOR. The parentheses are required.

ENTITY REFERENCE ADVISOR1 (INSTRUCTOR),  
ADVISOR2 (ASST\_INSTRUCTOR)

Because the diagram contains a loop, the pair of variables can be repeated any number of times.

## Example 5



### Sample Input

PS MODIFY 11159

### Explanation

The constants PS and MODIFY are followed by the variable 11159, which is a request number.

PS MODIFY 11159,11160,11163

Because the diagram contains a loop, the variable 11159 can be followed by a comma, the variable 11160, another comma, and the final variable 11163.

PS MOD 11159-11161 DESTINATION =  
"LP7"

The constants PS and MODIFY are followed by the user-selected variables 11159-11161, which are request numbers, and the user-selected variable DESTINATION = "LP7", which is a file attribute phrase. Note that the constant MODIFY has been abbreviated to its minimum allowable form.

PS MOD ALL EXCEPTIONS

The constants PS and MODIFY are followed by the user-selected constants ALL and EXCEPTIONS.

# Glossary

## A

### **accesscode**

An identification code, subordinate to a usercode, which sometimes has an associated password of its own. An accesscode can be specified in the USERDATAFILE as required along with a usercode/password combination. An accesscode further establishes a user's identity, controls security, and restricts access to disk files.

### **ACU**

See automatic calling unit.

### **address**

(1) The identification of a location in storage (memory). (2) A sequence of bits, a character, or a group of characters that identifies a network station or a group of stations, a user, or an application.

### **ALGOL**

Algorithmic language. A structured, high-level programming language that provides the basis for the stack architecture of the Unisys A Series systems. ALGOL was the first block-structured language developed in the 1960s and served as a basis for such languages as Pascal and Ada. It is still used extensively on A Series systems, primarily for systems programming.

### **area**

The amount of contiguous disk space that is allocated at one time to a disk file as it is being created or expanded. *Synonym for row.*

### **ASCII**

American Standard Code for Information Interchange. A standard 7-bit or 8-bit information code used to represent alphanumeric characters, control characters, and graphic characters on a computer system.

### **attached station**

A station under the control of a message control system (MCS).

### **automatic calling unit (ACU)**

A device that permits a modem or data terminal to place calls automatically and thereby establish a dialed link over the communication network. This device is used with data circuit terminating equipment (DCE) to connect line adapters to data communications lines.

### B

#### backup file

(1) A printer or punch file assigned to a backup peripheral for subsequent output. The default backup peripheral is a disk. (2) A copy of a file that is stored offline so that it can be copied back onto the system if the original file becomes corrupted or inaccessible. (3) A copy of a file on a cataloging system that has been saved with one of the following Work Flow Language (WFL) statements: *COPY & BACKUP*, *ARCHIVE DIFFERENTIAL*, *ARCHIVE FULL*, *ARCHIVE INCREMENTAL*, or *ARCHIVE ROLLOUT*.

#### BACKUPPROCESS

A task initiated with the Command and Edit (CANDE) BACKUPPROCESS command that enables the user to list, identify, copy, or remove backup files.

#### beginning of job (BOJ)

The start of processing of a job.

#### beginning of task (BOT)

The start of processing of a task.

#### bit

The most basic unit of computer information. The word *bit* is a contraction of *binary digit*. A bit can have one of two values: binary 0 (sometimes referred to as OFF) and binary 1 (sometimes referred to as ON).

#### block

(1) A group of physically adjacent records that can be transferred to or from a physical device as a group. (2) A program, or a part of a program, that is treated by the processor as a discrete unit. Examples are a procedure in ALGOL, a procedure or function in Pascal, a subroutine or function in FORTRAN, or a complete COBOL program.

#### BLOCKSIZE

A file attribute that gives the length of a file block.

#### BNA

The network architecture used on A Series, B 1000, and V Series systems as well as CP9500 and CP 2000 communications processors to connect multiple, independent, compatible computer systems into a network for distributed processing and resource sharing.

#### BOJ

See beginning of job.

#### Boolean item

In Data Management System II (DMSII), a data item that stores information having a logical value of TRUE or FALSE.

#### borrowed station

A station that one message control system (MCS) recognizes as belonging to another MCS.

**BOT**

See beginning of task.

**buffer**

An area in which data is stored temporarily.

**byte**

(1) (ANDIPS) A binary character string operated upon as a unit and usually shorter than a computer word. (2) On Unisys A Series systems, a measurable group of 8 consecutive bits having a single usage. In data communications, a byte is often referred to as a character or an octet.

**C**

**CANDE**

See Command and Edit.

**CANDE/CODE file**

An unsaved object code file produced by a Command and Edit (CANDE) compilation. This file is titled CANDE/CODE <recovery number>, where <recovery number> is a number assigned by CANDE to identify the file.

**CANDE/RECV file**

A recovery file created by the Command and Edit (CANDE) message control system (MCS) that contains any work file changes since the last update, as well as the title and other attributes of that work file. This file is titled CANDE/RECV <recovery number>, where <recovery number> is a number assigned by CANDE to identify the file.

**CANDE/TEXT file**

A Command and Edit (CANDE) work file that has been updated but not yet saved. This file is titled CANDE/TEXT <recovery number>, where <recovery number> is a number assigned by CANDE to identify the file. When the CANDE *SAVE* command is entered, the title of the work file is changed to the title under which the user obtained the work file by using the CANDE *GET* or *MAKE* command.

**central processing unit (CPU)**

The computer hardware unit that controls and executes the instructions contained in object code files.

**character**

(1) The actual or coded representation of a digit, letter, or special symbol in display form.  
(2) In data communications, 8 contiguous bits (1 byte).

**checksum**

A directory test that performs a data integrity check of a directory record to ensure that it has not been corrupted.

**circuit**

A transmission medium connecting two or more electronic devices. A circuit is the configuration of equipment used in transmitting data from one location to another and can involve more than one type of facility.

## Glossary

---

### **code file**

*See* object code file, source file.

### **Command and Edit (CANDE)**

A time-sharing message control system (MCS) that enables a user to create and edit files, and to develop, test, and execute programs, interactively.

### **compiler**

A computer program that translates instructions written in a source language, such as COBOL or ALGOL, into machine-executable object code.

### **continuation character**

In the Command and Edit (CANDE) message control system (MCS), a character entered at the end of a line of terminal input to allow that line to be continued on the next line.

### **control command**

A Command and Edit (CANDE) input message that begins with a specific character, usually a question mark (?), and is used to control or interrogate the CANDE operating environment. A control command can be entered from any attached CANDE station.

### **control station**

In the Command and Edit (CANDE) message control system (MCS), a station that allows CANDE network control commands to be entered.

### **CONTROLLER**

An invisible, independent runner program that is responsible for processing system commands, routing system messages, and scheduling jobs.

### **CPU**

*See* central processing unit.

### **CREATIONDATE**

The file attribute that gives the date on which a file was first entered into a directory on a disk.

### **crunch**

To return the unused portion of the last row of disk space of a disk file (beyond the end-of-file indicator) to the system. Only disk files can be crunched.

### **CTRL (control) key**

A special function key on some keyboards that generates control sequences for transmission to the system.

### **cursor**

The marker on the terminal screen that indicates where the next character entered appears.



**D****data circuit terminating equipment (DCE)**

In X.25, the functional unit of a data station that establishes, maintains, and releases a connection and provides the functions necessary for any code or signal conversion between the data terminal equipment (DTE) and the data transmission line. A DCE can be a separate piece of equipment. A DCE is the network supplier's equipment that can serve several user installations and is the user's entry point to the network.

**data comm**

*See* data communications.

**data communications (data comm)**

The transfer of data between a data source and a data sink (two computers, or a computer and a terminal) by way of one or more data links, according to appropriate protocols.

**Data Communications ALGOL (DCALGOL)**

A Unisys language based on ALGOL that contains extensions for writing message control system (MCS) programs and other specialized system programs.

**data communications file**

A logical file that allows communication with a remote device.

**Data Management System II (DMSII)**

A specialized system software package used to describe a database and maintain the relationships among the data elements in the database.

**DCALGOL**

*See* Data Communications ALGOL.

**DCE**

*See* data circuit terminating equipment.

**disabled**

Referring to a station in which messages from the line it represents are being ignored by the system.

**distributed systems service (DSS)**

One of a collection of services provided on Unisys hosts to support communications across multihost networks. DSSs can be services such as file handling, station transfer, and mail transfer.

**DMSII**

*See* Data Management System II.

**DSS**

*See* distributed systems service.

**DSS Management**

A collection of functions provided to facilitate installation and functioning of distributed systems services (DSSs). For example DSS Management can initiate the appropriate

## Glossary

---

DSS provider upon receipt of a service request and reinitiate a DSS provider after a halt/load.

### **DSS provider**

A library or application program that supplies a function that qualifies as a distributed systems service (DSS).

## **E**

### **EBCDIC**

Extended Binary Coded Decimal Interchange Code. An 8-bit code representing 256 graphic and control characters that are the native character set of most mainframe systems.

### **enabled**

Referring to a station that is being polled (invited to transmit in a certain order) and that can communicate with the system.

### **end of job (EOJ)**

The termination of processing of a job.

### **EOJ**

See end of job.

### **external file name**

The name used to identify a physical file, which is a file on the system rather than a file declared in a program. The external file name is accessed through the FILENAME and TITLE file attributes.

## **F**

### **family name**

(1) The name, consisting of up to 17 alphanumeric characters, assigned by an installation to identify a family of disks. (2) The name (label) of the disk or disk pack on which a physical file is located. The family name of a file is determined by the value of the FAMILYNAME file attribute. (3) The name of the logical group of disk packs on which a physical file is located. A family name consists of from 1 to 17 alphanumeric characters and is assigned by the installation.

### **file attribute**

An element that describes a characteristic of a file and provides information the system needs to handle the file. Examples of file attributes are the file title, record size, number of areas, and date of creation. For disk files, permanent file attribute values are stored in the disk file header.

### **File Transfer Protocol (FTP)**

A protocol used to copy disk files between hosts connected across a Transmission Control Protocol/Internet Protocol (TCP/IP) network.

**File Transfer, Access, and Management (FTAM)**

The standard developed by the International Standards Organization (ISO) for file exchange and management across an Open Systems Interconnection (OSI) network. FTAM systems can access file attributes (for example, password information) and the contents of files (including individual records, as well as entire files). *See also* OSI File Transfer, Access, and Management.

**FTAM**

*See* File Transfer, Access, and Management.

**FTP**

*See* File Transfer Protocol.

**G****guard file**

A disk file created by the GUARDFILE utility program that describes the access rights of various users and programs to a program, data file, or database.

**H****halt/load**

A system-initialization procedure that temporarily halts the system and loads the master control program (MCP) from a disk to main memory.

**Host Services**

A collection of services that are provided across multihost networks of A Series, B 1000 Series, CP9500, and V Series systems. The services can include file transfer, job transfer, station transfer, logical I/O, remote tasking, and remote command entry. *Contrast with* DSS Management and Network Services.

**I****I/O**

Input/output. An operation in which the system reads data from or writes data to a file on a peripheral device such as a disk drive.

**I/O processor (IOP)**

A specialized processor for moving data between system memory and the I/O subsystem.

**I/O subsystem**

The hardware and software that manage all transfers of information between the operating system and peripheral devices.

**ID field**

A group of columns in a file record that usually contain descriptive information about the record.

## Glossary

---

### **input queue**

The Command and Edit (CANDE) queue containing commands that are entered at a station while CANDE is still executing a previous command from that station. While the previous command is executing, the input queue is said to be in normal state. The first 20 entries in the input queue are known as the visible portion of the queue.

### **IOP**

See I/O processor.

## **J**

### **job**

An independent process. The job of a particular task is the independent process that is the eldest ancestor of that task.

## **K**

### **KIND**

The file attribute that indicates the type of device on which the file is stored.

## **L**

### **line**

(1) A row of text in a printout. (2) A data transmission link between two computers or between a computer and its associated terminals. (3) In X.25, the portion of a data circuit external to the data circuit terminating equipment (DCE) that connects the data terminal equipment (DTE) to an exchange or to one or more DCEs, or connects two exchanges.

### **line support processor (LSP)**

The data communications subsystem processor that manages communication with the host and initiates processes that control the input of messages to and the output of messages from data communications lines.

### **literal (mode) search**

In the Command and Edit (CANDE) message control system (MCS) and the Editor, a search mode in which each line of the file is considered to be a string of characters (including blanks). The search for target text is successful whenever a sequence of characters in a line of the file exactly matches the target text. The number of blanks separating strings of alphanumeric or nonblank graphic characters in the target text and in the line of the file must be the same. The other search mode used by CANDE and the Editor is token search.

### **log off**

The process by which a user ends the current message control system (MCS) session.

### **log on**

The process by which a user starts a message control system (MCS) session. The user identifies himself or herself as a legitimate user of the system by entering a valid

usercode/password and, in some cases, a valid chargecode and accesscode. (In addition, an accesscode can have an associated password of its own.)

**log station**

A Command and Edit (CANDE) station designated to receive logging information. This information can include statistics on station attachment, security errors, station log on and log off times, network changes caused by reconfiguration requests, and user messages sent to the operator display terminal (ODT).

**logical station number (LSN)**

A unique number assigned to each station in a network and each pseudostation allocated by a message control system (MCS). Each station has an LSN assigned according to the order in which the stations are defined.

**LSN**

See logical station number.

**LSP**

See line support processor.

**M****MARKID**

In the Command and Edit (CANDE) message control system (MCS) and the Editor, the value placed in the ID field of new or modified records.

**MAXINPUT**

In the Command and Edit (CANDE) message control system (MCS), a parameter that determines the number of characters a terminal can transmit to the host at one time.

**MAXOUTPUT**

In the Command and Edit (CANDE) message control system (MCS), a parameter that determines the number of characters a terminal can receive from the host and display at one time.

**MAXRECSIZE**

A file attribute that gives the maximum length, in frames, of records in a logical file. For port files, MAXRECSIZE specifies the maximum text size for all subfiles in the port file.

**MCS**

See message control system.

**MERGE**

The Command and Edit (CANDE) command that copies a file into the work file by collating the sequence numbers from the two files. If a record in the copied file and a record in the work file have the same sequence number, the work file record is preserved.

**message**

(1) Any combination of characters and symbols designed to communicate information from an originator to one or more destinations. (2) The text sent to the user from a

## Glossary

---

program. A message can be either displayed on the screen or printed. (3) In data communications, any information-containing data unit, in an ordered format, sent by means of a communications process to a named network entity or interface. A message contains the information (text portion) and controls for routing and handling (header portion).

### **message control system (MCS)**

A program that controls the flow of messages between terminals, application programs, and the operating system. MCS functions can include message routing, access control, audit and recovery, system management, and message formatting.

### **message level interface processor (MLIP)**

See I/O processor.

### **MINRECSIZE**

A file attribute that gives the minimum length, in frames, of records in a logical file.

### **MLIP**

An acronym for the obsolete term message level interface processor; see I/O processor.

### **modem**

A device placed between a computer system and a telephone line to permit transmission of digital pulses. Modems permit computers to communicate with other computers, terminals, and printers over communication lines. The term *modem* is derived from *modulator-demodulator*.

## N

### **Native File Transfer (NFT)**

A distributed systems service (DSS) that provides file transfer between A Series hosts across a BNA network.

### **NDLII**

See Network Definition Language II.

### **Network Definition Language II (NDLII)**

The Unisys language used to describe the physical, logical, and functional characteristics of the data communications subsystem to network support processors (NSPs), line support processors (LSPs), and data communications data link processors (DCDLPs).

### **network information file II (NIFII)**

The file generated when a Network Definition Language II (NDLII) program is compiled. This file contains line support processor (LSP) and network support processor (NSP) code, data structures, and other information. A NIFII is also generally referred to as a network information file (NIF).

### **Network Services**

In BNA, the component responsible for formatting, routing, and transmitting messages. In BNA Version 1, Network Services is divided into components at three logical levels: the Port Level, the Router Level, and the Station Level. In BNA Version 2, Network Services is also divided into components at three logical levels: the Port Level, the

Network Layer (which includes routing), and the Link Layer (plus its companion Physical Layer). Network Services is under the control of the network services manager (NSM).

**network support processor (NSP)**

A data communications subsystem processor that controls the interface between a host system and the data communications peripherals. The NSP executes the code generated by the Network Definition Language II (NDLII) compiler for line control and editor procedures. An NSP can also control line support processors (LSPs).

**NFT**

See Native File Transfer.

**NIFII**

See network information file II.

**NSP**

See network support processor.

## O

**object code file**

A file produced by a compiler when a program is compiled successfully. The file contains instructions in machine-executable object code.

**object directory**

A directory of the files stored under *OBJECT/<node name>* that are object code files. Object code files are usually associated with corresponding source files.

**object file**

See object code file.

**object program**

A set or group of executable machine-language instructions and other material designed to interact with data to provide problem solutions. An object program is generally the machine-language result of the operation of a high-level language compiler on a source program. See also object code file, compiler.

**ODT**

See operator display terminal.

**Open Systems Interconnection (OSI)**

A set of data communications standards defined by the International Organization for Standardization (ISO) that provide for communications between different types of computer systems. The application services defined under OSI include File Transfer, Access, and Management (FTAM) and the Message Handling System (MHS).

**operator display terminal (ODT)**

(1) A terminal or other device that is connected to the system in such a way that it can communicate directly with the operating system. The ODT allows operations personnel to accomplish system operations functions through either of two operating modes:

## Glossary

---

system command mode or data comm mode. (2) The name given to the system control terminal (SCT) when it is used as an ODT.

### OSI

See Open Systems Interconnection.

### OSI File Transfer, Access, and Management (OSI FTAM)

An A Series distributed systems service (DSS) that supports Open Systems Interconnection (OSI) standards for file management functions such as reading, writing, and copying files on remote hosts.

### OSI FTAM

See OSI File Transfer, Access, and Management.

## P

### page mode

In the Command and Edit (CANDE) message control system (MCS), the mode in which the user can enter and edit one screen of text at a time. The commands that invoke page mode are the PAGE, NEXT, SAME, SEQ, +, and - commands.

### password

A character string associated with a usercode or accesscode in the USERDATAFILE, and used to identify legitimate users of the system. When logging on to a message control system (MCS), a user must supply a usercode and a password.

### pending state

The state of the Command and Edit (CANDE) input queue after the original executing command has completed. At that time, a CANDE ?GO, ?WAIT, ?PURGE, ?TAKE, or ?ENTER command can be entered to manipulate the queue, or the queued input can be discarded by entering any other CANDE command.

### private file

A file with a SECURITYTYPE attribute specified as PRIVATE. Only users logged on to a privileged usercode or to the usercode under which the file is stored can access a private file.

### process

(1) The execution of a program or of a procedure that was initiated. The process has its own process stack and process information block (PIB). It also has a code segment dictionary, which can be shared with other processes that are executions of the same program or procedure. (2) A software application; that is, any activity or systematic sequence of operations that produces a specified result.

### public file

A file with a SECURITYTYPE attribute specified as PUBLIC. Users who are logged on to any usercode can access a public file by specifying the (<usercode>)<file name> form for the file title.



**R****record**

(1) A group of logically related items of data in a file that are treated as a unit. (2) The data read from or written to a file in one execution of a read or write statement in a program.

**recovery file**

File used to recover a work file in the event that a Command and Edit (CANDE) session or an Editor session on a particular station is aborted.

**remote job entry (RJE)**

A Unisys message control system (MCS) that allows jobs, data, and control commands to be sent to a central system from a remote card reader; RJE also allows output of data from the central system to be sent to remote peripherals.

**RJE**

See remote job entry.

**RMERGE**

A Command and Edit (CANDE) command that copies a file into the work file, collating the sequence numbers from the two files. Whenever a record in the copied file and a record in the work file have the same sequence number, the record in the copied file is preserved.

**run time**

The time during which an object code file or user interface system (UIS) is executed. *Synonym for* execution time and, in COBOL, object time.

**run-time error**

An error occurring during the execution of a program, which causes the system software to terminate execution of that program abnormally.

**S****saved text queue**

The Command and Edit (CANDE) queue where each normal line of input — that is, each line that does not contain a control command — is stored after it is entered. Entries in the saved text queue can be examined, edited, and subsequently reexecuted.

**schedule file**

A file containing all Command and Edit (CANDE) commands and programs entered for a schedule session.

**schedule session**

A Command and Edit (CANDE) session that runs independently of the user's session and station, and executes all CANDE commands and program input that are present in a previously specified schedule file.

## Glossary

---

### **schedule station**

A dummy data communications station provided by the Command and Edit (CANDE) message control system (MCS) to run schedule sessions and open remote files. The schedule station behaves programmatically like a real station for most purposes.

### **secured-terminal environment**

An environment in which every user of a terminal on a message control system (MCS) is required to supply the MCS with an acceptable usercode and password and, possibly, with an acceptable accesscode (with or without an associated password).

### **SECURITYTYPE**

The file attribute that specifies the type of access allowed to a file.

### **sequence field**

In the Editor, a portion of each line that is defined for use as a location in which to store line-sequencing information.

### **sequence mode**

In the Command and Edit (CANDE) message control system (MCS), the mode initiated by the SEQ command in which the system interprets any entered text except control commands and certain forms of the CANDE *FIX* and *MARGIN* commands as new text to be added to the work file. The system automatically assigns sequence numbers to all lines that are interpreted as work file text.

### **session**

(1) The interactions between a user and a message control system (MCS) during a particular period of time that is assigned an identifying session number. Logging on initiates a new session; logging off terminates a session. Each Menu-Assisted Resource Control (MARC) or Command and Edit (CANDE) dialogue at a terminal accesses a different session. (2) In the Command and Edit (CANDE) message control system (MCS), the time measured from when a CANDE user enters a valid usercode/password combination until that user enters a CANDE *SPLIT*, *BYE*, or *HELLO* command. Each CANDE session is assigned a unique number. No output is printed until the session is ended.

### **source file**

(1) A file in which a source program is stored. (2) A file containing instructions written in a programming language.

### **source program**

A program coded in a language that must be translated into machine language before execution. The translator program is usually a compiler.

### **SPO**

(1) An acronym for the obsolete term supervisory printer output; see operator display terminal. (2) In Network Definition Language II (NDLII), an attribute used to confer or remove control station status.

### **stack**

A region of memory used to store data items in a particular order, usually on a last-in, first-out basis. *Synonym for process stack.*

**station**

(1) The outer end of a communication line. A station can correspond to a single terminal connected on a single line, or several stations can be connected on a line. (2) The combination of functional units comprising the data terminal equipment (DTE), data circuit terminating equipment (DCE), and their common interface. (3) In data communications, a data structure that relates a logical abstraction to either a physical device or a pseudostation.

**supervisory printer output (SPO)**

An obsolete term; *see* operator display terminal.

**synchronous transmission**

In data communications, a mode of data transmission in which each bit of data is transmitted at a frequency determined by an external clock source.

**T****tankfile**

In the Command and Edit (CANDE) message control system (MCS), a file maintained to store option settings, configuration information, and work file recovery information (including all changes made since the work file was last updated). The tankfile is used for recovery if CANDE terminates abnormally.

**target text**

A string of text specified in a Command and Edit (CANDE) FIND command, which the find operation attempts to locate in each line of a file.

**task**

(1) A dependent process. (2) Any process, whether dependent or independent. *See also* process.

**terminal**

An I/O device designed to receive or send source data in a network.

**TITLE**

A file attribute whose value is the external name of a file. By default, this value is the value of the INTNAME attribute. For a file whose KIND attribute is equal to DISK or PACK, the TITLE attribute can be assigned a value of the form *<file name> ON <family name>*; thus, the values of the TITLE and FILENAME attributes, both of which specify the external file name, can be different.

**token (mode) search**

In the Command and Edit (CANDE) message control system (MCS) and the Editor, a search mode in which each line of the file is considered to be a sequence of tokens, where a token is considered to be any string of alphanumeric or special characters that does not include blanks. The search for target text is successful whenever the sequence of tokens in the target text matches a sequence of tokens within a line in the file. In effect, the number of blanks separating strings in the target text is ignored. *Contrast with* literal (mode) search.

## Glossary

---

### transmit key

See XMIT.

### turnaround

(1) The time required for a job to be completed. In dealing with job queue maintenance, turnaround time is the time that has elapsed since the last job was introduced into the mix from a particular queue. (2) In data communications, the operation of reversing the direction of transmission from send to receive or receive to send. Turnaround time is the time required to perform this operation.

## U

### universal file name

A file name that satisfies normal A Series naming conventions or that satisfies the naming conventions of a host that is not an A Series host.

### usercode

An identification code used to establish user identity and control security, and to provide for segregation of files. Usercodes can be applied to every task, job, session, and file on the system. A valid usercode is identified by an entry in the USERDATAFILE.

### USERDATAFILE

A system database that defines valid usercodes and contains various data about each user (such as accesscodes, passwords, and chargecodes) and the population of users for a particular installation.

## W

### waiting state input queue

The state of the Command and Edit (CANDE) input queue after a *?WAIT* control command has been entered and any CANDE operation currently in progress has finished. When the input queue is in waiting state, the user must enter a *?PURGE* control command to clear the input queue or a *?GO* or *?REPEAT* control command to execute the queued commands. Any CANDE command other than *?GO*, *?REPEAT*, or *?PURGE* is placed at the end of the input queue when the queue is in waiting state.

### WFL

See Work Flow Language.

### WFL job

(1) A Work Flow Language (WFL) program, or the execution of such a program. (2) A collection of Work Flow Language (WFL) statements that enable the user to run programs or tasks.

### word

A unit of computer memory. On A Series systems, a word consists of 48 bits used for storage plus tag bits used to indicate how the word is interpreted.

**work file**

A file that the user accesses using the Command and Edit (CANDE) GET command or creates using the CANDE *MAKE* command. All editing commands entered through CANDE or the Editor can change only the current work file.

**Work Flow Language (WFL)**

A Unisys language used for constructing jobs that compile or run programs on A Series systems. WFL includes variables, expressions, and flow-of-control statements that offer the programmer a wide range of capabilities with regard to task control.

**X**

**XMIT (transmit) key**

The key on the terminal keyboard that, when pressed, transmits the entered data to the computer system. *Synonym for* XMT (transmit) key.

**Z**

**zero suppression**

The elimination of insignificant zeros during a printing operation.



# Bibliography

- A Series ALGOL Programming Reference Manual, Volume 1: Basic Implementation* (form 8600 0098). Unisys Corporation.
- A Series ALGOL Test and Debug System (TADS) Programming Guide* (form 1169539). Unisys Corporation.
- A Series CANDE Configuration Reference Manual* (form 8600 1344). Unisys Corporation.
- A Series COBOL ANSI-74 Programming Reference Manual, Volume 1: Basic Implementation* (form 8600 0296). Unisys Corporation.
- A Series Communications Management System (COMS) Operations Guide* (form 8600 0833). Unisys Corporation.
- A Series Distributed Systems Service (DSS) Operations Guide* (form 8600 0122). Unisys Corporation.
- A Series File Attributes Programming Reference Manual* (form 8600 0064). Unisys Corporation. Formerly *A Series I/O Subsystem Programming Reference Manual*.
- A Series FORTRAN77 Test and Debug System (TADS) Programming Guide* (form 3950 8759). Unisys Corporation.
- A Series I/O Subsystem Programming Guide* (form 8600 0056). Unisys Corporation. Formerly *A Series I/O Subsystem Programming Reference Manual*.
- A Series Print System (PrintS/ReprintS) Administration, Operations, and Programming Guide* (form 8600 1039). Unisys Corporation.
- A Series Printing Utilities Operations Guide* (form 8600 0692). Unisys Corporation.
- A Series Security Administration Guide* (form 8600 0973). Unisys Corporation.
- A Series Security Features Operations and Programming Guide* (form 8600 0528). Unisys Corporation.
- A Series System Commands Operations Reference Manual* (form 8600 0395). Unisys Corporation.
- A Series System Software Support Reference Manual* (form 8600 0478). Unisys Corporation.
- A Series System Software Utilities Operations Reference Manual* (form 8600 0460). Unisys Corporation.

## Bibliography

---

*A Series Task Attributes Programming Reference Manual* (form 8600 0502). Unisys Corporation. Formerly *A Series Work Flow Administration and Programming Guide*.

*A Series Work Flow Language (WFL) Programming Reference Manual* (form 8600 1047). Unisys Corporation.



# Index

## A

A Series Native File Transfer (NFT), 3-41  
ACCESS command, 3-2  
accesscode, 1-1  
    changing, 3-2, 3-5  
    interrogating, 3-2  
    recovery files used with, 1-4  
ACCESSCODE task attribute, 1-5  
<accesscode>, 3-2  
activity level, displaying CANDE  
    ?COUNTS control command, 4-18  
ADD command, 3-4, 3-27  
<after specification>, 3-194  
<alphanumeric character>, 2-1  
ALTER command, 3-4A  
    file attributes, 3-4A, 3-4B  
<alternate family>, 3-62  
APASSWORD command, 3-5  
<apassword>, 3-2  
AREAS attribute, 1-9  
AREASIZE attribute, 1-9  
ASDU control command, 4-2  
AT control command, 4-3  
    FTAM option, 4-3  
    HOSTSERVICES option, 4-3  
    HS option, 4-3  
<attribute option>, 3-83  
attributes, file  
    limitations of, 1-9  
attributes, of CANDE file, 1-9  
AX control command, 4-11

## B

backup files  
    copying, 3-7  
    printing of, 3-194  
BACKUP option, COPY command, 3-31  
Backup Processor utility, 3-7  
BACKUPPROCESS command, 3-7  
BANNER option, 3-194

<base>, 2-1  
BIND command, 3-10  
BLOCKSIZE attribute, 1-9  
<Boolean expression>, COPY command,  
    3-33  
break condition, 1-6  
BREAKPOINT control command, 4-12  
BRK, 1-6  
BYE command, 3-11

## C

C control command, 4-13  
CANDE  
    attribute limitations, 1-9  
    commands, 3-1  
    file attributes, 1-9  
CANDE control commands, 4-1  
CATALOG option, COPY command, 3-31  
CHANGE command, 3-12  
    ?AT control command option, used for  
        FTAM file management, 4-3  
CHANGE command, LOCKEDFILE file  
    attribute, 3-13  
<change files>, 3-12  
<change from files>, 3-12  
<change from group>, 3-12  
CHAR control command, 4-14  
character  
    changing end-of-line  
        ?EOL control command, 4-29  
    redefining special  
        ?CHAR control command, 4-14  
<character>, 2-1  
CHARGE command, 3-17  
chargecode, 1-1  
    changing, 3-17  
<chargecode>, 3-17

## Index

---

- CHECKPOINT option, 3-194
  - CODE file, 1-10
    - <code file task attribute>, 3-18
    - <code visibility mnemonic>, 2-1
  - column indicator, 1-14
    - <column part>, 3-122
    - <column selection>, 3-122
    - <column>, 2-1
  - commands
    - CANDE, 3-1
      - using with the SCHEDULE command, 3-149
    - control, 4-1
      - deleting from the queue, 4-72
      - display commands in queues, 4-61
      - entering, 1-14
      - functional groupings, 5-1
      - multiple, 1-5
      - response to, 1-7
      - resuming processing, 4-32
    - COMMENT control command, 4-16
    - COMPARE option, COPY command, 3-30
    - comparing files, 3-101
    - COMPILE command, 3-18
    - compiler
      - invoking, 3-18
      - task status
        - ?CS control command, 4-19
    - <compiler name>, 2-1
    - <compiler specification>, 3-18
    - <compiler type>, 2-1
    - COMS window dialogs, 1-1
    - CONNECT
      - command, 3-21
      - control command, 4-17
    - CONNECT control command, 4-17
    - constructs, basic, 2-1
    - CONTINUE command, 3-23
      - <control char mnemonic>, 2-1
    - control commands, 4-1
    - control transfer
      - of data comm station, 4-17
      - one MCS to another, 3-105
    - CONVENTION command, 3-25
    - COPIES option, 3-194
    - COPY command, 3-27
      - BACKUP option, 3-31
      - CATALOG option, 3-31
      - COMPARE option, 3-30
        - <copy file>, 3-32
        - <copy from group>, 3-33
        - <copy request>, 3-32
      - <destination volume>, 3-33
      - file attributes, 3-34, 3-35, 3-36
      - FROMSTART option, 3-31
        - <FTAM transform attribute list>, 3-33
      - FTAM transform attributes, 3-38, 3-39, 3-40
        - <FTP transform attribute list>, 3-33
      - FTP transform attributes, 3-37
        - <log-on info>, 3-33
        - <source volume>, 3-33
        - <transfer service> option, 3-31
      - VERIFY option, 3-31
    - <copy file>, 3-27
      - COPY command, 3-32
    - <copy from group>, 3-28
      - COPY command, 3-33
    - <copy request>, 3-27
      - COPY command, 3-32
  - copying
    - backup files, 3-7
    - lines from a file, 3-80
  - core utilization for a task
    - ?CU control command, 4-20
  - <count>, 3-66
  - COUNTS control command, 4-18
  - CS control command, 4-19
  - CU control command, 4-20
- D**
  - data comm network commands, functional grouping, 5-2
  - data comm station, transferring control
    - ?CONNECT control command, 4-17
  - data comm subsystem status, 3-50
  - date, displaying
    - TD control command, 4-74
    - TIME control command, 4-77
    - WT control command, 4-87
  - <date>, 3-149, 3-166
  - <day interval>, 3-167
  - DCPREFIX, displaying
    - ?ID control command, 4-36
  - DCSTATUS command, 3-50
  - default file attributes, 1-9
  - DELETE command, 3-52
  - deletion, single-line, 3-161
  - DENY control command, 4-21
    - <depth>, 3-64
    - <dest file name>, 3-66
    - <destination volume attribute list>, 3-29

< destination volume > , 3-28  
     COPY command, 3-33  
 < destination > , 3-194  
 DESTNAME command, 3-53  
 dialogs, COMS window, 1-1  
 < digit > , 2-1  
 < directory name > , 2-1  
 < directory title > , 2-2  
 DISCARD command, 3-55  
 display file contents, 3-86  
 < dlm > , 2-2  
 DO command, 3-56  
 document types, FTAM, 3-43  
 documents, related, vii  
 DOUBLE option, 3-194  
 DS control command, 4-22  
 DUMP control command, 4-23  
 duplicate libraries, removal  
     ?RM control command, 4-56

## E

ECHO function, 4-24  
 edit commands, functional grouping, 5-1  
 EDIT control command, 4-25  
 < edit specs > , 3-71  
 elapsed time, displaying  
     ?TI control command, 4-76  
 < end column > , 2-2  
 END control command, 4-26  
 end-of-file commands, functional grouping,  
     5-2  
 ending current session  
     BYE command, 3-11  
     HELLO command, 3-78A  
 ENTER control command, 4-27  
 entering  
     commands, 1-14  
     text, 1-14  
 entry, single-line, 3-161  
 EOL control command, 4-29  
 ESCAPE command, 3-58A  
 EXCEPTIONEVENT, causing  
     ?HI control command, 4-33  
 EXCLUDE command, 3-59  
 EXECUTE command, 3-60

## F

FA control command, 4-30

FAMILY command, 3-62  
 < family name > , 2-2  
 < family specifications > , 3-62  
 family substitution, 1-6  
 file  
     accessing, 3-76  
     adding, 3-27  
     attributes, 1-9  
     backup, copying, 3-7  
     changing security attributes, 3-153  
     CODE, 1-10  
     comparing, 3-101  
     compiling, 3-166  
     copying, 3-27  
     copying lines from, 3-80  
     creating a work file, 3-92  
     default attributes, 1-9  
     display contents, 3-86  
     end-of-file signaling, 4-26  
     file attribute information, 3-83  
     intrinsic file, displaying, 4-62  
     line range, displaying, 3-126  
     listing  
         on cards, 3-194  
         on printer, 3-194  
     listing names of files, 3-64  
     moving lines within, 3-108  
     news, listing, 3-111  
     recovery, 1-10  
         accesscodes and, 1-4  
         recalling as a work file, 3-129  
         removal, 3-55  
     removing, 3-131  
     resequencing line numbers, 3-140  
     restart, 1-3  
     saving, 3-147  
     searching for text within, 3-66  
     startup, 1-3  
     system log file, 3-91  
     TEXT, 1-10  
     updating work file, 3-183  
     work file, (See work file)  
 < file attribute/print modifier assignment > ,  
     3-119  
 file attributes  
     in COPY command, 3-34  
     in LFILES command  
         used with ?AT control command, 4-6  
 < file directory > , 2-2  
 < file management command > , 4-3  
 < file mnemonic primary > , COPY command,  
     3-33

## Index

---

<file name>, 2-2  
<file title>, 2-2  
File Transfer Protocol (FTP), 3-41  
File Transfer, Access, and Management (FTAM), 3-42  
    document types, 3-43  
    file transfer restrictions, 3-46  
    service for file access, 4-5  
FILEKIND attribute, changing, 3-181  
FILENAME option, 3-194  
FILES command, 3-64  
    ?AT control command option, used for FTAM file management, 4-3  
FIND command, 3-66  
FIND command, hexadecimal escape character, 3-66  
FIX command, 3-71  
FIX command, hexadecimal escape character, 3-72  
FORMID option, 3-194  
FR control command, 4-31  
FROMSTART option, COPY command, 3-31  
FTAM, (See File Transfer, Access, and Management (FTAM))  
    <FTAM file attributes>, 4-4  
    FTAM option, ?AT control command, 4-3  
    <FTAM text>, 4-3  
    <FTAM transform attribute list>, 3-28  
        COPY command, 3-33  
    FTAM transform attributes, 3-32  
        COPY command, 3-38, 3-39, 3-40  
FTP, (See File Transfer Protocol (FTP))  
    <FTP transform attribute list>, 3-28  
        COPY command, 3-33  
    FTP transform attributes, 3-32  
        COPY command, 3-37  
    FTPDATA file, 3-44  
    FTPDATA, FILEKIND, 3-41  
functional command groupings, 5-1

## G

GET command, 3-76  
GO control command, 4-32  
<guard file title>, 3-153

## H

HELLO command, 3-78A

hexadecimal escape character, FIND command, 3-66  
hexadecimal escape character, FIX command, 3-72  
hexadecimal escape character, REPLACE command, 3-136  
HEXPLICIT option, 3-194  
HI control command, 4-33  
HN control command, 4-35  
host  
    connecting to, 3-21  
    displaying hostnames, 4-35  
    remote, directing information to  
        ?ASDU control command, 4-2  
        ?AT control command, 4-3  
Host Services File Transfer, 3-41  
Host Services for file access, 4-3  
    <hostname>, 2-2  
hostnames, displaying  
    ?HN control command, 4-35  
<hyphen>, 2-3

## I

I/O time, displaying  
    ?TI control command, 4-76  
<ID char>, 2-3  
ID control command, 4-36  
identification, user, 1-1  
<identifier>, 2-3  
<inc>, 2-3  
information, related product, vii  
initiating  
    new session, 3-78A  
        SPLIT command, 3-165  
    page mode operation, 1-13, 3-112, 3-114, 3-146, 3-203  
    program dump, 4-23  
input  
    continuing lines, 3-23  
    purging  
        ?PURGE control command, 4-50  
    queue manipulation, 1-7  
input editing commands, functional grouping, 5-2  
<input file title>, 3-149  
INSERT command, 3-80  
<integer expression>, COPY command, 3-33  
<integer>, 2-3  
<interchange file name>, 4-4

<interchange name>, 3-28  
 <interchange string constant>, 4-3  
 intrinsics file, displaying  
   ?SI control command, 4-62  
 invoking  
   compiler, 3-18  
   LOGANALYZER, 3-91  
   page mode operation, 3-203  
   WFL compiler, 3-166, 3-192

## J

JA control command, 4-37  
 jobs  
   information on  
     ?MXA control command, 4-42  
   list scheduled  
     ?S control command, 4-58  
   list waiting  
     ?W control command, 4-80  
   listing completed  
     ?C control command, 4-13  
   queued list  
     ?SQ control command, 4-65  
   start time  
     ?STARTTIME control command, 4-68  
   status  
     ?JA control command, 4-37  
 <julian date>, 3-166

## K

KIND attribute, 1-9

## L

LANGUAGE command, 3-82  
 <language name>, 3-82  
 <letter>, 2-3  
 LFILES command, 3-83  
   ?AT control command option, used for  
     FTAM file management, 4-3  
 libraries, displaying list of frozen  
   ?LIBS control command, 4-38  
 library commands, functional grouping, 5-1  
 LIBS control command, 4-38  
 limitations of CANDE files, 1-9  
 <line width>, 3-149

line, single entry or deletion, 3-161  
 lines  
   continuing input  
     CONTINUE command, 3-23  
   copying from a file, 3-80  
   deleting, 3-52  
   moving, 3-108  
   range, displaying, 3-126  
   resequencing numbers, 3-140  
 LIST command, 3-86  
 LOAD command, 3-76  
 LOCKEDFILE file attribute, CHANGE  
   command, 3-13  
 LOG command, 3-91  
 <log-on account>, 3-30, 4-3  
 log-on attempts, maximum, 1-2  
 <log-on info>, 3-29, 4-3  
 <log-on info>, COPY command, 3-33  
 <log-on password>, 3-30, 4-3  
 <log-on usercode>, 3-30, 4-3  
 LOGANALYZER, invoking, 3-91  
 logging on, 1-1  
 <LSN range>, 2-3  
 <LSN>, 2-3

## M

MAKE command, 3-92  
 manuals, related, vii  
 MARGIN command, 3-95  
 MARKID command, 3-97  
 master control program (MCP), displaying  
   current name  
     ?WM control command, 4-84  
 MATCH command, 3-101  
 <match output options>, 3-101  
 MAXRECSIZE attribute, 1-9  
 MCS, (See message control system (MCS))  
   command, 3-105  
   control command, 4-39  
 memory management parameters, displaying  
   values  
     ?SF control command, 4-60  
     ?TF control command, 4-75  
 memory modules, displaying status  
   ?MM control command, 4-40  
 MERGE command, 3-106  
 message control system (MCS)  
   transferring control to and from, 3-105,  
     4-39  
 messages

## Index

---

sending  
  from a scheduled session, 4-52  
  SS control command, 4-66  
  TO control command, 4-78

task  
  ?MSG control command, 4-41

MINRECSIZE attribute, 1-9

minus (-) command, 3-203

<mix number list>, 2-3

<mix number>, 2-3

MM control command, 4-40

mnemonic values  
  for print attributes, 3-123  
  for print modifiers, 3-123

modules, displaying status of memory  
  ?MM control command, 4-40

MOVE command, 3-108

moving  
  commands in the save queue, 4-55  
  lines from one file to another, 3-80  
  within a work file, 1-13, 3-112, 3-114

MSG control command, 4-41

multiple commands, 1-5

MXA control command, 4-42

## N

<name>, 2-4

<named parameter list>, 3-166

names of files, listing, 3-64

Native File Transfer (NFT), 3-41

<new apassword>, 3-5

<new directory name>, 3-12

<new file name>, 3-12

<new file>, 3-101

NEWS command, 3-111

news file, listing, 3-111

NEXT command, 1-13, 1-14, 3-112

NF control command, 4-43

NFT, (See Native File Transfer (NFT))

<NLS>, 2-4

NOHEADING option, 3-194

nondata file, transferring through Host Services, 3-41

nondata file, transferring through NFT, 3-41

<nonquote EBCDIC character>, 2-4

<nonsingle quote EBCDIC character>, 2-4

NORESTART control command, 4-53

NOTE option, 3-194

NOTOK control command, 4-45

<number>, 2-4

NUMBERED control command, 4-46

NUMBERED option, 3-194

## O

object program, executing, 3-60

OF control command, 4-47

OK control command, 4-48

<old apassword>, 3-5

<old file>, 3-101

Open Systems Interconnection (OSI) FTAM,  
  3-42

<option list>, 2-4

OSI, (See Open Systems Interconnection  
  (OSI) FTAM)

OT control command, 4-49

<output file name>, 3-149

<output option>, 3-83

## P

PAGE command, 1-13, 1-14, 3-114

page mode commands, functional grouping,  
  5-1

page mode operation, 1-12

  column indicator, 1-14

  entering

    commands, 1-14

    text, 1-14

  initiation, 1-13

  moving through the work file, 1-13

  requirements, 1-12

PAGECOMP file attribute, 3-124A

  examples of syntax, 3-124C

<parameters>, 3-166

password, 1-1

  changing, 3-116

PASSWORD command, 3-116

<password>, 2-5

PDEF command, 3-119

plus (+) command, 3-203

<positional parameter list>, 3-166

print

  specifying station, 3-53

<print attribute phrase>, 3-122

print attributes

  mnemonic values for, 3-123

PRINT command, 3-121

  PAGECOMP file attribute, 3-124A

PRINTPARTIAL file attribute, 3-124A  
 TRANSFORM file attribute, 3-124B  
 < print default assignment list >, 3-121  
 < print mnemonic >, 3-122  
 < print modifier phrase >, 3-122  
 print modifiers  
     mnemonic values for, 3-123  
 < print specification >, 3-121  
 < print-partial string >, 3-122  
 < printcharge >, 3-194  
 PRINTDEFAULTS command, 3-119  
 < printdisposition >, 3-194  
 PRINTERCONTROL option, 3-194  
 < printerkind >, 3-194  
 printing  
     backup files, 3-194  
 PRINTPARTIAL file attribute, 3-124A  
 process time, displaying  
     ?TI control command, 4-76  
 product information, vii  
 program dump, initiating  
     ?DUMP control command, 4-23  
 pseudostations, 1-1  
 PURGE control command, 4-50

## Q

< qe# >, 4-27  
 QIC tape, (See quarter inch cartridge (QIC) tape)  
 quarter inch cartridge (QIC) tape, 3-30  
 queue  
     input queue manipulation, 1-7  
     saved text queue manipulation, 1-7  
 queue managing commands, functional  
     grouping, 5-2  
 < queue number >, 4-65  
 queued input  
     purging  
         ?PURGE control command, 4-50  
     suspending, 4-80A

## R

railroad diagrams, explanation of, A-1  
 RANGE command, 3-126  
 reading data from tape, 3-176  
 record  
     changing the value, 3-97

    deleting in page mode, 3-190  
 record formats (table), 2-8, 2-9  
 < record selection >, 3-122  
 < record/line part >, 3-122  
 RECOVER command, 1-11, 3-129  
 recovery files, 1-10  
     accesscodes and, 1-4  
     recalling as a work file, 3-129  
     removal, 3-55  
 < recovery number >, 1-10  
 recovery, resuming input line processing,  
     4-54  
 refreshing displayed page  
     SAME command, 3-146  
 remote file, 1-12  
 remote host, directing information to  
     ?ASDU control command, 4-2  
     ?AT control command, 4-3  
 REMOVE command, 3-131  
     ?AT control command option, used for  
         FTAM file management, 4-3  
 < remove file list >, 3-131  
 < remove file >, 3-131  
 < remove from group >, 3-131  
 RENEW command, 3-134  
 REPEAT control command, 4-51  
 REPLACE command, 3-135  
 REPLACE command, hexadecimal escape  
     character, 3-136  
 REPORT control command, 4-52  
 requests, denying program  
     ?DENY control command, 4-21  
 RESEQ command, 3-140  
 response to CANDE commands, 1-7  
 RESTART control command, 4-53  
 restart file, 1-3  
 RESUME control command, 4-54  
 RETRIEVE control command, 4-55  
 RM control command, 4-56  
 RMERGE command, 3-142  
 RO  
     command, 3-163  
     control command, 4-64  
 RUN command, 3-60

## S

S control command, 4-58  
 SAME command, 1-13, 1-14, 3-146  
 SAVE command, 3-147  
 SAVEBACKUPFILE option, 3-194

## Index

---

- saved text
  - editing
    - ?EDIT control command, 4-25
  - queue manipulation, 1-7
- SAVEFACTOR attribute, 1-9
- SC control command, 4-58A
- SCHEDULE
  - command, 3-149
  - control command, 4-59
- schedule control commands, functional grouping, 5-2
- search commands, functional grouping, 5-1
- searching for text within a file, 3-66
- SECURITY command, 3-153
  - < security file >, 3-153
  - < security from group >, 3-153
  - < security specification >, 3-153
- SECURITYTYPE attribute, 1-9
  - < securitytype >, 3-194
- SECURITYUSE attribute, 1-9
  - < securityuse >, 3-194
- SEQ command, 1-13, 1-14, 3-156
  - < sequence number >, 2-5
- sequence numbers
  - causing to appear, 4-46
  - suppressing, 4-46
  - < sequence part >, 3-122
  - < sequence range list >, 2-5
  - < sequence range >, 2-5
  - < serial number list >, COPY command, 3-33
- session
  - commands, functional grouping, 5-1
  - inquiry, 3-172
  - options, setting and resetting, 3-163
  - restarting
    - ?RESTART control command, 4-53
  - starting another, 3-165
  - status report, schedule
    - ?SCHEDULE control command, 4-59
  - terminating, 3-165
  - terminating scheduled sessions, 3-173
- SF control command, 4-60
- SHOW control command, 4-61
- SI control command, 4-62
- single-line entry or deletion, 3-161
- SL control command, 4-63
- SO
  - command, 3-163
  - control command, 4-64
  - < source volume attribute list >, 3-29
  - < source volume >, 3-28
  - COPY command, 3-33
  - < special character >, 2-5
- SPLIT command, 3-165
- SQ control command, 4-65
- SQUASHED option, 3-194
- SS control command, 4-66
- ST control command, 4-67
- stack cell, interrogating contents
  - ?OT control command, 4-49
- < standard compiler >, 2-5
- < start column >, 2-6
- START command, 3-166
- STARTTIME control command, 4-68
  - < starttime spec >, 3-166
- startup files, 1-3
- station
  - identification
    - ?WRU control command, 4-85
  - LSN
    - ?WHERE control command, 4-82
  - name
    - ?WD control command, 4-81
    - ?WHERE control command, 4-82
  - status
    - ?STATUS control command, 4-69
    - ?STUP control command, 4-71
  - < station name >, 2-6
- STATUS
  - command, 3-172
  - control command, 4-69
- status of data comm subsystem, 3-50
  - < ste# >, 4-55
- STOP command, 3-173
  - < string >, 2-6
- STUP
  - control command, 4-71
  - < substitute family >, 3-62
- substitution, family, 1-6
- suspended task, cancel processing
  - ?NOTOK control command, 4-45
- suspended task, resuming processing
  - ?OK control command, 4-48
- < switch >, 2-6
- system
  - configuration, displaying
    - ?SC control command, 4-58A
  - inquiry commands, functional grouping, 5-2
  - log file, displaying, 3-91
  - supervisor program, displaying name
    - ?CS control command, 4-19
    - ?WS control command, 4-86



## T

TAB command, 3-174  
 TAKE control command, 4-72  
 TAPE command, 3-176  
 tape, specifying final reel  
   ?FR control command, 4-31  
 < target family >, 3-62  
 < target specs >, 3-135  
 task  
   cancel processing  
     ?NOTOK control command, 4-45  
   causing EXCEPTIONEVENT  
     ?HI control command, 4-33  
   commands, functional grouping, 5-1  
   core utilization return  
     ?CU control command, 4-20  
   information on, 4-83  
   interrogation and control commands,  
     functional grouping, 5-2  
   list scheduled  
     ?S control command, 4-58  
   list waiting  
     ?W control command, 4-80  
   listing completed  
     ?C control command, 4-13  
   resuming processing  
     ?OK control command, 4-48  
   status  
     ?CS control command, 4-19  
     ?STATUS control command, 4-69  
     ?STUP control command, 4-71  
   suspending  
     ?ST control command, 4-67  
   terminating  
     ?DS control command, 4-22  
 < task attribute assignment >, COPY  
   command, 3-33  
 < task equation list >, 2-7  
 TD control command, 4-74  
 TERMINAL command, 3-178  
 terminal information commands, functional  
   grouping, 5-2  
 terminating  
   current session, 3-78A  
   current session and starting another,  
     3-165  
   scheduled sessions, 3-173  
   task, 4-22  
 terminating a CANDE session  
   BYE command, 3-11  
 text

editing  
   ?EDIT control command, 4-25  
 entering, 1-14  
 replacing  
   REPLACE command, 3-135  
 TEXT file, 1-10  
 TF control command, 4-75  
 TI control command, 4-76  
 TIME control command, 4-77  
 < time interval >, 3-167  
 time, displaying  
   current  
     TD control command, 4-74  
     TIME control command, 4-77  
     WT control command, 4-87  
   elapsed, 4-76  
   I/O, 4-76  
   process, 4-76  
 < time >, 3-166  
 TITLE  
   command, 3-12  
   option, 3-194  
 TO control command, 4-78  
 < trainid >, 3-194  
 < transfer service >, 3-27  
 < transfer service > option, COPY command,  
   3-31  
 transferring control  
   data comm station, 4-17  
   to another MCS, 3-105  
 transferring MCS control  
   ?MCS control command, 4-39  
 TRANSFORM file attribute, 3-124B  
 < transform >, 3-194  
 TRUNCATED option, 3-194  
 TYPE command, 3-181  
 < type >, 2-7

## U

< underscore >, 2-7  
 UNITS attribute, 1-9  
 < universal change list >, 4-3  
 < universal file name >, 3-28  
 < universal file title >, 4-3  
 UNNUMBERED control command, 4-46  
 UNSEQUENCED option, 3-194  
 UPCASED option, 3-194  
 UPDATE command, 3-183  
 user  
   identification, 1-1

## Index

---

information commands, functional  
grouping, 5-1  
USERBACKUPNAME option, 3-194  
usercode, 1-1  
<usercode>, 2-7  
USERDATAFILE, 1-1  
UTILITY command, 3-185  
utility commands, functional grouping, 5-1

## V

VERIFY option, COPY command, 3-31  
version, displaying current CANDE  
?WRU control command, 4-85  
<visibility mnemonic>, 2-7  
VOID command, 3-190

## W

W control command, 4-80  
WAIT control command, 4-80A  
WD control command, 4-81  
WFL command, 3-192  
WHAT command, 3-193  
WHERE control command, 4-82  
WHY control command, 4-83  
window dialogs, COMS, 1-1  
WM control command, 4-84  
work file, 1-10  
changing name, 3-12  
commands, functional grouping, 5-1  
compiling, 3-166  
copying lines into, 3-80  
creating, 3-92  
creating from existing file, 3-76  
deleting lines from, 3-52  
interrogating state, 3-193  
moving lines within, 3-108  
moving within, 1-13, 3-112, 3-114  
recalling a recovery file as, 3-129  
removing, 3-131  
resequencing line numbers, 3-140  
saving, 3-147  
updating, 3-183  
Work Flow Language (WFL) compiler, 3-192  
WRITE command, 3-194  
<write options>, 3-194  
WRU control command, 4-85  
WS control command, 4-86

WT control command, 4-87

## Y

Y control command, 4-83

?% (Comment) control command, 4-16

Publication Title \_\_\_\_\_

Form Number \_\_\_\_\_

Unisys Corporation is interested in your comments and suggestions regarding this manual. We will use them to improve the quality of your Product Information. Please check type of suggestion:

 Addition Deletion Revision Error

Comments: \_\_\_\_\_

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

Name \_\_\_\_\_ Telephone number \_\_\_\_\_

Title \_\_\_\_\_ Company \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip code \_\_\_\_\_

Cut along dotted line ✂

Tape

Please Do Not Staple

Tape

Fold Here



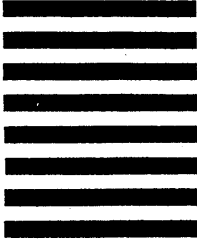
NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

**BUSINESS REPLY MAIL**

FIRST CLASS MAIL PERMIT NO. 817 DETROIT, MI

POSTAGE WILL BE PAID BY ADDRESSEE

UNISYS CORPORATION  
ATTN: PUBLICATIONS  
25725 JERONIMO ROAD  
MISSION VIEJO, CA 92691-9826







86001500-000