Rocket UniVerse

Guide to the UniVerse Editor

Version 11.3.1

October 2016
UNV-1131-EDIT-1

Notices

Edition

Publication date: October 2016
Book number: UNV-1131-EDIT-1
Product version: Version 11.3.1

Copyright

Trademarks

Examples

This information might contain examples of data and reports. The examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

License agreement

Note: This product may contain encryption technology. Many countries prohibit or restrict the use, import, or export of encryption technologies, and current use, import, and export regulations should be followed when exporting this product.

Corporate information

Rocket Software, Inc. develops enterprise infrastructure products in four key areas: storage, networks, and compliance; database servers and tools; business information and analytics; and application development, integration, and modernization.

Website: www.rocketsoftware.com

Rocket Global Headquarters
77 4th Avenue, Suite 100
Waltham, MA 02451-1468
USA

To contact Rocket Software by telephone for any reason, including obtaining pre-sales information and technical support, use one of the following telephone numbers.

Country	Toll-free telephone number
United States	1-855-577-4323
Australia	1-800-823-405
Belgium	0800-266-65
Canada	1-855-577-4323
China	800-720-1170
France	08-05-08-05-62
Germany	0800-180-0882
Italy	800-878-295
Japan	0800-170-5464
Netherlands	0-800-022-2961
New Zealand	0800-003210
South Africa	0-800-980-818
United Kingdom	0800-520-0439

Contacting Technical Support

The Rocket Customer Portal is the primary method of obtaining support. If you have current support and maintenance agreements with Rocket Software, you can access the Rocket Customer Portal and report a problem, download an update, or read answers to FAQs. To log in to the Rocket Customer Portal or to request a Rocket Customer Portal account, go to www.rocketsoftware.com/support.

In addition to using the Rocket Customer Portal to obtain support, you can use one of the telephone numbers that are listed above or send an email to support@rocketsoftware.com.

Contents

Chapter 1: Using the UniVerse editor

This chapter describes how to use the UniVerse Editor. It includes explanations and examples of the most commonly used Editor commands, which perform functions such as the following:

- Provide online help
- Insert new lines in a record
- Display one or more lines on the screen
- Move the current line pointer
- Search for character strings or patterns
- Change or delete lines
- Exit the current record
- Define and manipulate blocks of lines
- Store and reexecute Editor commands

The UniVerse Editor is a line editor. You use it to add, change, or delete records in a UniVerse file. You can also use the UniVerse Editor to create and change UniVerse BASIC source programs, which are stored as records in a nonhashed type 1 or type 19 file. The file must exist before you can use the Editor to add new records or modify existing records.

Invoking the UniVerse Editor

To invoke the UniVerse Editor, use one of the following commands at the system prompt:

Command	Description
`ED`	Invokes the Editor and prompts for the name of the file. If select list 0 is active, the Editor loads the selected records in order after you enter the file name. If select list 0 is not active, the Editor prompts for a record.
`ED` *filename*	Invokes the Editor on *filename*. If select list 0 is active, the Editor loads the selected records in order. If select list 0 is not active, the Editor prompts for a record.
`ED` *filename records*	Invokes the Editor on specified records in *filename*. You can specify more than one record. If select list 0 is active, the Editor processes the records specified by the select list first and then the explicitly specified records.
`ED` *filename* *	Invokes the Editor on all records in a file.

The Editor works on one record at a time. You must specify record IDs, or an asterisk (*) to specify all records in the file, unless a select list is active. You can specify both new records and existing records in the same command.

Note: The Editor does not allow empty record IDs. Do not confuse empty with null value. The empty string is a character string of zero length which is known to have no value. Null's value is unknown.

How the UniVerse Editor works

The Editor lets you edit one record in a file at a time. It displays each field of the record on a separate line. Each line is identified by a four-digit number by default.

```
0001: JAMES
```

When you invoke the Editor on an existing record, a message shows how many lines make up the record:

```
>ED VOC TEST
5 lines long.
```

If the record specified by the ED command does not exist, the Editor creates a new record and displays a new record prompt:

```
----:
```

You can enter Editor commands in response to this prompt. Editor commands let you change the contents of the current line, reposition the current line pointer, display some or all of the lines, insert new lines of text, and manipulate blocks of lines. To execute an Editor command, enter the command and press Enter.

The UniVerse Editor is a line editor. The Editor maintains a current line pointer that keeps track of the position of the current line. Each command you enter at the Editor prompt usually affects only the current line or a range of lines starting with the current line (the exception is when you are working with predefined blocks of lines). When you first invoke the Editor on a record, it positions the current line pointer at the top of the record just before the first line.

```
>ED VOC TEST
5 lines long. ----:
```

Editor command processing

Each Editor command is pushed onto an Editor command stack, which is similar to the command processor sentence stack. Any commands in the stack can be changed and reexecuted using a set of stack commands similar to the command processor stack commands. One or more Editor commands can be stored in the special Editor file, &ED&, or another specified file for later execution.

For more information about the Editor command stack and stored Editor commands, see Using the editor stack commands, on page 21.

Editor commands

Most Editor commands take the form of a single letter followed by an argument. Some Editor commands use a numeric argument to specify a line number or a number of lines. Other commands use a string or pattern to specify characters on a line.

The following sections discuss the most commonly used Editor commands.

Using command line help

Use the HELP command to obtain a summary of the Editor commands and their syntax. To get help, enter one of the following:

HELP

HELP *string*

If you enter HELP by itself, a prompt appears. Press Enter to display the descriptions for all the Editor commands.

string can be a letter or series of characters that identify the set of commands you want to display. If you enter `HELP` followed by a letter, all the Editor commands that begin with that letter are displayed.

Enter `HELP C` to display the following information:

```
----: HELP C
C       - Do the last 'CHANGE' command again.
C///            - CHANGE one or more lines. formats permitted are:
                            C/from/to       C/from/to/#
                            C/from/to/G     C/from/to/#G    C/from/to/G#
                            C/from/to/B     C/from/to/BG    C/from/to/GB
        where   / - is any delimiter character.
                    from - is the character string to be replaced.
                        to - is the character string to substitute.
                         # - is the number of lines to CHANGE. (The default is one)
                         G - is the letter 'G' (global) CHANGE all instances in line.
                         B - is the letter 'B', CHANGE all lines in the defined BLOCK.
CAT any         - CONCATENATE the next line onto the current line,
                          separated by 'any'.
COL             - Display relative COLUMN POSITIONS on the CRT.
COPY            - COPY a BLOCK (see '<'and '>' ), source block is unchanged.
Top.
----:
```

If you enter `HELP` followed by a string, `HELP` locates commands whose descriptions contain that string. This format is useful when you know the function of a command, but not its name. Enter `HELP EXIT` to display the Editor commands whose description contains `EXIT`:

```
----: HELP EXIT
EX      - EXIT the editor (same as QUIT).
Q           - QUIT - EXIT the editor.
QUIT        - QUIT - EXIT the editor.
X           - EXIT (QUIT) from the editor and abandon an active SELECT list.
Top.
----:
```

To spool the `HELP` output to a printer, use the `SPOOLHELP` command.

Inserting new lines

Two Editor commands insert new lines into a record: `I` and `IB`. Use the Insert (`I`) command to insert one or more lines of text after the current line. Use the Insert Before (`IB`) command to insert lines of text before the current line. The syntax is as follows:

I [*text*]

IB [*text*]

If you do not specify *text*, the Editor prompts you to insert lines you want to add. The Editor prompt changes to *nnnn*= where *nnnn* is the line number of the line you are about to enter. Here is an example:

```
0001=
```

If Insert is the first command given when a record is opened, a new line is inserted at the top of the record. After you enter text on the line, press Enter to insert the new line. When you finish inserting lines, press Enter at the beginning of the next blank line. This returns you to the Editor prompt. To insert a blank line, enter a space at the beginning of the line.

Notice that IB followed by a single space as *text* on the command line does not store an empty line before the current line. The Insert (I) command followed by a single space on the command line does store an empty line at the top of the record.

The following example demonstrates how to use the Insert (I) command to insert eight new lines of data in a new record:

```
>ED DISTRIBUTORS ER3120
New record.
-----= I
0001= JAMES
0002= SANDERS
0003= 131 CRESTVIEW
0004= PIEDMONT
0005= NY
0006= SNOWMOBILES
0007= ACME SALES
0008= 1960
0009= <Return>
Bottom at line 8.
 ----:
```

Displaying specified lines

Several Editor commands display lines in the current record. The commands for displaying specified lines are as follows:

Command	Description
L [*lines*]	Displays the next line or the next specified number of lines.
P [*lines*]	Prints 23 lines or the specified number of lines.
PL [*lines*]	Prints the next 20 lines or the specified number of lines.
PL [*–lines*]	Prints the previous 20 lines or the specified number of lines.
PP [*lines*]	Prints 10 lines (or the specified number of lines) before and after the current line.
<Enter>	Displays the next line.

Using the List (L) command

The List (L) command displays the next line. If you include a numeric argument with the L command, the current line and the specified number of lines after it are displayed, just as with the Print (P) command.

If an L command with a string argument (the Locate command) was entered during the current editing session, L without an argument repeats the previous Locate command instead of displaying the next line.

Using the Print (P) command

By default, the Print (P) command displays the next 23 lines and moves the current line pointer to the last line displayed. You can specify the number of lines you want to display as an argument to the P command.

Using the Print Lines (PL) command

By default the Print Lines (`PL`) command displays the next 20 lines of the record but does not move the current line pointer. You can change the default to display the number of lines you want to display either before or after the current line. Use a positive number to display lines after the current line; use a negative number to display lines before the current line.

Using the Print Page (PP) command

Use the Print Page (`PP`) command to display 10 lines before and 10 lines after the current line. Change the default by specifying the total number of lines (not including the current line) you want to display.

Pressing return

Press Enter to display the line following the current line. When you press Enter at the bottom of a record, the Editor moves the current line pointer to the top of the record. <Enter>within an example indicates that the Enter key was pressed.

Examples of line display commands

The following examples show how to use the display commands:

```
>ED TEST DISPLAY
81 lines long.
----: P6
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
0006: LINE6
----: <Return>0007: LINE7
----: <Return>0008: LINE8
----: L0009: LINE9
----: L0010: LINE10
----: L30010: LINE10
0011: LINE11
0012: LINE12
----: PL-110001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
0006: LINE6
0007: LINE7
0008: LINE8
0009: LINE9
0010: LINE10
0011: LINE11
0012: LINE12

0012: LINE12
----:
----: PP60
009: LINE9
0010: LINE10
0011: LINE11
0012: LINE12
```

```
0013: LINE13
0014: LINE14
0015: LINE15
0012: LINE12
----:
```

Moving the current line pointer

The Editor has a number of commands that move the current line pointer to another position. If you press Enter, the current line pointer moves to the next line in the record. The other commands that move the pointer are as follows:

Command	Action
line#	Line number
B	Bottom
F	Find
G	Go to
L[*string*]	Locate
M	Match
PO	Position
T	Top

Using the Go to (G) command

The Line number (*line#*) and Go to (G) commands are similar. Use them to specify the line you want to make the current line. You can enter the number of the line by itself or enter `Gline#`. The pointer moves to a new current line.

```
----: G1
0001: JAMES
----: G4
0004: PIEDMONT
```

Using the Position (PO) command

You can use the Position (PO) command with a line number to move to a line. If you enter PO without a line number, the pointer moves to the top of the record.

```
----: PO6
0006: SNOWMOBILES
----: POTop.
```

Using the Top (T) and Bottom (B) commands

Use the Top (T) and Bottom (B) commands to move the pointer to the top or the bottom of the record. For example:

```
----: T
Top.
----: B
0008: 1960
```

```
Bottom at line 8.
```

Using the search commands

The Find (F), Locate (L), and Match (M) commands are all search commands. Each command searches the record for the next line containing a specified string or pattern and then displays the line. The search commands have the following syntax:

F [[*col#*] *string*]

L [*string*]

M [*pattern*]

The following guidelines describe when to use each command:

- Use Find (F) to search for a string by its column location in a line. If a Find is not executed, the current line pointer moves to the next line.
- Use Locate (L) to search for the next line that contains a specific string of characters.
- Use Match (M) to search for a line that matches a specified pattern.

When you use the M *pattern* command, the entire line must match the entire pattern.

You can, however, use M to match a pattern that is part of a line if you use three dots (…) as follows:

Pattern	Description
...pattern	Line must end with *pattern.*
*pattern****...***	Line must begin with *pattern.*
...pattern...	Line can contain *pattern* anywhere in the line.
*pattern1****...****pattern2*	Line must begin with *pattern1* and end with *pattern2.*

The following examples demonstrate how to use the Locate, Match, and Find commands:

```
----: 2
0002: SANDERS
----: L MO
0004: PIEDMONT
----: L0
006: SNOWMOBILES
----: L
Bottom at line 8.
----: T
Top.
----: M SANDERS
0002: SANDERS
----: M 131
Bottom at line 8.
----: T
Top.
----: 131...
0003: 131 CRESTVIEW
----: F N
0005: NY
----: T
Top.
----: F5 C
0003: 131 CRESTVIEW
```

In the previous example, typing `F5 C` finds the next line with C in the fifth position by specifying the column number.

Changing and deleting lines

The following Editor commands let you change, delete, break, join, or duplicate lines in a record:

Command	Description
A	Append
B	Break
C	Change
CAT	Concatenate
D or DE	Delete line
DUP	Duplicate
OOPS	Undo last change command
R	Replace

Changing and replacing lines

There are two commands for changing lines in a record: the Change (C) command and the Replace (R) command. Use the Change or Replace command to replace an old string of characters with a new string. Use the Replace command to replace the entire line with a new line.

Syntax

The syntaxes of the two commands are the same:

C/*string* [/ [*new.string*]] [/ [**G**] [**B** | *lines*]]

R/*string* [/ [*new.string*]] [/ [**G**] [**B** | *lines*]]

string is the string of characters that is to be replaced by *new.string*. *lines* specifies the number of lines after the current line that you want to change. The G (Global) option specifies that *string* be changed each time it appears in the specified *lines.* The B (Block) option lets you specify a block of lines in which to effect the change. You must define a block of lines before you can use the B option.

Delimiters are shown as slashes, but they can be any of the following characters:

Delimiter characters						
!	@	#	$	%	&	*
/	\	:	=	+	-	
(	)	{	}	[	]	
‘	’	.	|	"	,	

The Replace command also allows the following syntax:

R [*new.string*]

When you use the Replace command without delimiters, you have only to type the replacement string, not the old string.

If you enter C or R without arguments, the last C or R command is reexecuted. The simplest form of the Change command is as follows:

C/*string*/*new.string*

Use this syntax to change a string of characters on the current line only. To delete a string in more than one line, use the following syntax:

C/*string*//[**G**] *lines*

The following example demonstrates how to use the Change command. Change finds string number 60 in the current line and replaces it with the new string number 82. Change then deletes the string REPRESENTATIVE from the current line. Notice the space included at the beginning of the string. This space ensures that no trailing space is left at the end of the line after the deletion.

```
----: 8
0008: 1960
----: C/60/82
0008: 1982
----: 7
0007: ACME SALES REPRESENTATIVE
----: C/ REPRESENTATIVE
0007: ACME SALES
----:
```

The following example shows how you can use quotation marks as delimiters to change a line containing a slash character (/):

```
0011: 4/4
----: C"/4"/2"
0011: 4/2
```

Suppose you want to change the variable CHAR to NUMBER in the first six lines of your program. Position the pointer at the top of the record and use the Change command to search six lines to find the string CHAR and change all instances of it to NUMBER. The G makes sure that every occurrence of CHAR on a line is changed, as in lines 5 and 6 where it occurs three times.

```
----: T
Top.
----: C/CHAR/NUMBER/G6
0001: PRINT "ENTER A NUMBER":
0002: INPUT NUMBER1
0003: PRINT "ENTER ANOTHER NUMBER":
0004: INPUT NUMBER2
0005: CALL ADDEMUP(NUMBER1,NUMBER2,NUMBER3)
0006: PRINT NUMBER1:"PLUS":NUMBER2:"IS":NUMBER3
----:
```

The following example shows how to replace a single line in a record:

```
----: 2
0002: SANDERS
----: R BURTON
0002: BURTON
----:
```

Appending a character string to a line

Use the Append (A) command to append a string of characters to the end of the current line. The syntax is as follows:

A *string*

The A command does not have a global option. To append the same string to several lines, move to a new line and enter A again, without specifying a string. The Editor uses the string from the last Append command issued. Repeat as often as necessary.

The following example shows how to use the Append command. Note that there are two spaces between the A and MANAGER, so that the resultant line has a space between SALES and MANAGER.

```
----: 7
0007: ACME SALES
----: A  MANAGER
0007: ACME SALES MANAGER
----:
```

Deleting lines

Use the Delete Lines (D or DE) command to delete one or more lines from the current record. The syntax is as follows:

D[*lines*]**DE**[*lines*]

lines is the number of lines, beginning with the current line, that are to be deleted. After you delete the specified lines, the current line pointer returns to the line preceding the first deleted line.

Do not confuse the Delete Lines (D or DE) command with the DELETE command. D and DE delete only the specified lines of a record, whereas DELETE deletes the entire record from the file. The following example shows how to use the Delete Lines (D) command without specifying the number of lines to delete:

```
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
0006: LINE6
0007: LINE7
Bottom at line 7.
----: D
Bottom at line 6.
----: T
Top.
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
0006: LINE6
Bottom at line 6.
----: T
Top.
----: D2
----: P
0001: LINE3
0002: LINE4
0003: LINE5
0004: LINE6
Bottom at line 4.
----:
```

Undoing your last change

Use the OOPS command to undo the last change made to a record. Any command that modifies the record can be undone with OOPS. To restore a record, you must use OOPS before using another modify command. However, OOPS does not have to immediately follow the modify command. For example, you can use position commands to change the current line without affecting OOPS. Use the Inquire (?) command to see the last modify command that can be undone by OOPS.

Saving your changes

Use the SAVE command to save the current record and continue editing the same record. The syntax is as follows:

SAVE [[*filename*] *record*]

The SAVE command writes the record currently being edited to the file from which it was read. After you save the record, it remains open for further editing. If you specify a file name to save the current record, you must also specify a record name. If you specify only a record name, SAVE writes a copy of the record to the current file under its new name. The original record is also retained in the file under its old name. The following example shows how to use the SAVE command:

```
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
Bottom at line 4.
----: SAVE
"RECORD" filed in file "TEST".
Bottom at line 4.
----:
```

Exiting a record

The Editor provides several commands for exiting the current record. They let you exit and save your current changes, discard the current changes, or delete the current record. The commands are as follows:

Command	Description
DELETE or FD	Exits the current record, deleting it from the file.
EX or Q or QUIT	Exits the current record, discarding current changes.
FI or FILE	Files and exits the current record, saving current changes.
N	If a select list is active, exits the current record, discarding current changes, and displays the next selected record. The select list is not discarded.
X	If a select list is active, exits the current record, discarding current changes and the select list. X exits both the current record and the Editor, returning to the system prompt.

Saving your changes and exiting the record

Use the FI or FILE command to save the changes made to the current record. The syntax is as follows:

FI [[*filename*] *record*]

FILE [[*filename*] *record*]

If you do not specify *filename* or *record*, FILE saves the current record in the current file. Include *filename* when you want to save the current record in a file other than the current one. The file must already exist. If you specify *filename*, you must also specify *record* even if the name is to be the same as the current record.

If you specify only *record*, the Editor files it in the current file. The Editor informs you if the record already exists. To overwrite an existing record, answer Y to the message that the Editor displays. If you answer N , the FILE command is not executed. The Editor then prompts you for another record or returns you to the UniVerse prompt, depending on how you invoked the Editor. If a select list is active, or if you specified more than one record when you entered the Editor, the FILE command saves the current record and displays the next record.

The following example demonstrates how to use the FILE command to file a record and return to UniVerse:

```
>EDTEST
File name          = TEST
Record name = RECORD
5 lines long.
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
Bottom at line 4.
----: FILE
"RECORD" filed in file "TEST".
File name          = TEST
Record name = <Enter>
>
```

Instead of pressing Enter to exit the Editor, you can enter another record name and continue editing.

Exiting the record without saving changes

Use the EX or Q command to leave the Editor without saving the current changes. The syntax is as follows:

EX

Q or **QUIT**

EX or Q discards any changes made since the last Save command and saves the original record. The Editor verifies that you want to discard the changes. If you answer Y , the Editor exits without saving the current version of the record. If you answer N , you can save the current record.

If a select list is active, you can use the Next (N) command to exit the current record without saving changes. The Editor then displays the next selected record. If a select list is active, you cannot use the EX, Q, or N commands to exit the current record and the Editor to return to the UniVerse prompt. To do this, use the X (Exit Editor) command.

Deleting the current record

Use the DELETE or FD command to delete the current record. You cannot retrieve a deleted record. The Editor verifies that you want to delete the record. If you answer Y , the record is deleted and the Editor prompts you for another record or returns to the UniVerse prompt, depending on how you invoked the Editor. If you answer N , the record is not deleted and you remain in the Editor.

Note the distinction between the DELETE (and FD) command and the D (and DE) command: DELETE or FD deletes the entire current record, whereas D or DE deletes only the specified lines. You cannot recover a deleted record, but you can restore deleted lines with the OOPS command.

Defining and using blocks

The Editor lets you mark blocks, or multiple lines of text, so you can move, copy, or delete them as a single unit. The block commands are listed in the following table.

Command	Description
<	Defines the first line of a block. At the top of the record, the block setting is canceled.
>	Defines the last line of a block. At the top of the record, the block setting is canceled.
< >	Defines a single line as a block.
BLOCK	Enables or disables the block verification prompt.
C	With the B option, changes a string in a block.
COPY	Copies the block to the current location. The original block remains in its location so that there are two copies of the block in the record.
DROP	Deletes the lines in the current block.
G<	Goes to the first line of the current block.
G>	Goes to the last line of the current block.
MOVE	Moves the block to the current location and deletes it from its original location.
PB	Prints the currently defined block.
R	With the B option, replaces a string in a block.

Working with blocks is a simple matter of defining the beginning and end of the block, then using the Change, COPY, MOVE, and DROP commands to change character strings within the block, or copy, move, or delete the entire block. Here are some examples:

```
----: P
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
0007: ACME SALES
0008: 1960
Bottom at line 8.
----: 3
0003: 131 CRESTVIEW
----: <
Block FROM set to line 3.
----: 5
0005: NY
----: >
Block THROUGH set to line 5.
----: G<
0003: 131 CRESTVIEW
----: 6
```

```
0006: SNOWMOBILES
----: COPY
BLOCK from 3 through 5. OK (Y) Y0007: 131 CRESTVIEW
----: T
Top.
----: P
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
0007: 131 CRESTVIEW
0008: PIEDMONT
0009: NY
0010: ACME SALES
0011: 1960
Bottom at line 11.
----: PB
0005: 131 CRESTVIEW
0006: PIEDMONT
0007: NY
0011: 1960
Bottom at line 11.
----: DROP
BLOCK from 3 through 5. OK (Y) Y
0002: SANDERS
----: T
Top.
----: P
0001: JAMES
0002: SANDERS
0003: SNOWMOBILES
0004: ACME SALES
0005: 1960
Bottom at line 5.
```

Getting the status of the current record

Use the Inquire (`?`) command to display a status report of the current record. Inquire displays the names of the file and the current record, the current line number, the setting of the up-arrow mode, any blocks that are set up, and the last change command that `OOPS` can restore. The next example shows the information displayed by the Inquire command:

```
0003= 131 CRESTVIEW
----: ?
Account Name                    = julie
File name                       = TEST
Record name                     = RECORD
Current line number             = 3
Up-arrow display mode           = disabled
Command execution ABORT         = disabled.
BLOCK operation verification        = disabled.
NO BLOCK currently defined.
No Pre-stored command has been executed this session.
An OOPS command will restore record prior to Command "C/LINE2/LINE3/"
```

Use the SIZE command to display the size of the current record. SIZE displays the number of lines and bytes. The following example shows the use of the SIZE command:

```
----: T
Top.
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
0006: LINE6
0007: LINE7
0008: LINE8
Bottom at line 8.
----: SIZE
Size of "ER3120" is 8 lines and 40 bytes (characters).
```

Note: You can use the RELEASE command to make the current record available to other users while you are using the Editor.

Using the editor stack commands

The Editor creates its own command stack that works the same way as the UniVerse sentence stack. The stack saves the 99 most recently used Editor commands, dropping the 99th command when the 100th command is added to the beginning of the stack. When you are using the Editor, you can store and save Editor commands in the special Editor file, &ED&, or another specified file.

Note that an Editor command ending with a question mark (?) is placed on the command stack but is not executed. The following table lists the Editor stack commands.

Command	Description
LOOP [*start*] [*rep*]	Repeats commands of a stored command sequence *rep* number of times, starting with *start*. If you do not specify *start* or *rep*, 1 is used for each.
PAUSE	Interrupts execution of a stored command sequence. Use the .XR command to resume execution after a PAUSE command. Use the .XK command to cancel execution after a PAUSE command.
.A [*line#*] *string*	Appends text to the end of an Editor command in the stack. If you do not specify *line#*, *string* is appended to sentence.
.C [*line#*] /*old*/*new*	Changes *old* to *new* in the command in the stack. If you do not specify *line#*, the change is made to sentence 1.
.D [*file*] [*line# record*]	Deletes an Editor command from the stack or a record from the &ED& file (or other file if specified). If you do not specify *line#*, sentence 1 is deleted.
.I [*line#*] *string*	Inserts a new Editor command into the stack. If you do not specify *line#*, *string* is inserted as sentence 1.

Command	Description
.L [*file*] [*lines* | *record*] .L [*file*] *	Lists the Editor commands in the stack or the commands stored in a record in the &ED& file (or other file if specified). If you do not specify *lines*, commands 1 through 9 are listed. .L *file* * lists the names of the prestored command records in &ED& or *file* if specified.
.R [*file*] [*line#* | *record*]	Recalls an Editor command in the stack, making it sentence 1, or recalls a stored command record from the &ED& file (or other *file* if specified).
.S [*line#*] [*file*] *record* .S [*file*] *recordstart,end*	Saves one or more Editor commands in the stack to a record in the &ED& file (or *file* if specified) for later execution. The first syntax line saves all commands from *line#* to sentence 1 to *record*. The second syntax line saves all commands from *start* to *end* to *record* in &ED& (or *file* if specified).
.U [*line#*]	Converts an Editor command in the stack to uppercase. If you do not specify *line#*, sentence 1 is converted to uppercase.
.X [*file*] [*line#* | *record*]	Reexecutes a command in the Editor stack or the set of commands stored in a record in the &ED& file (or *file* if specified). If you do not specify *line#*, sentence 1 is reexecuted. If you specify *record*, the commands stored in the &ED& file (or *file*) record are loaded into the stack and then executed.
.XK	Cancels execution of a stored command sequence after a PAUSE command suspends execution.
.XR	Resumes execution of a stored command sequence after a PAUSE command suspends execution.

Several of these commands are executed by the Editor the same way as they are executed by the command processor. They are .A, .C, .D, .I, .L, .R, and .S.

The following example shows how to execute a command from the Editor command stack:

```
>ED RECORDS RECORD1
5 lines long.
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
Bottom at line 5.
----: T
Top.
----: I THIS IS A TEST
0001: THIS IS A TEST
----: B
0006: LINE5
Bottom at line 6.
----: .L
04 P
03 T
02 I THIS IS A TEST
01 B0006: LINE5
Bottom at line 6.
```

```
----: .X2
02 I THIS IS A TEST
0007: THIS IS A TEST
Bottom at line 7.
----: FI
"REC1" filed in file "RECORDS".
```

The following example shows how to execute a command stored in the &ED& file:

```
>ED RECORDS REC3
New record.
----: .L INSERT
        INSERT
001 E Prestored Command saved at 15:55:02 11 Apr 1994
002 I LINE1
003 I LINE2
004 I LINE3
005 I LINE4Top.
----: .X INSERT
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
***** End of prestored command execution.
0005: LINE5
Bottom at line 4.
----: T
Top.
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
Bottom at line 4.
----: Q
*** Record changed, OK to Quit (Y) ? N
Bottom at line 4.
----: FI
"REC3" filed in file "RECORDS".
```

Editing non-ASCII characters

You may want to edit non-ASCII characters in UniVerse records that you cannot see on the screen. The Editor has a special mode for editing these characters. It is called up-arrow mode because you use the up-arrow, or caret, character (^) to switch the mode on or off.

There are two types of undisplayable non-ASCII character:

- Control characters and UniVerse system delimiters such as field marks and value marks.
- Characters from languages that your terminal does not support when you use UniVerse in NLS (National Language Support) mode. For information about NLS, see the *UniVerse NLS Guide*.

The next two sections describe the two flavors of up-arrow mode that correspond to these two types.

Editing system delimiters

Normally when you use the P (Print) command to list a record containing fields with several values, a character is substituted for the value marks themselves, since they cannot be displayed. For example:

```
Top
----: P
0001: Heitmann
0002: Andrea
0003: BAÿPhD
Bottom at line 3.
```

In field 3, the character ÿ represents the value mark (the actual character used varies from system to system). When up-arrow mode is enabled, the decimal character value represents the value mark and looks like this:

```
Top
----: ^
Up arrow display mode          = enabled
----: P
0001: Heitmann
0002: Andrea
0003: BA^253PhD
Bottom at line 3.
```

You can append another multivalue to field 3 of this record like this:

```
0003: BA^253PhD
----: A^253FRCO
0003: BA^253PhD^253FRCO
```

Editing in Unicode mode

When you use UniVerse with NLS enabled, you can use the Editor in Unicode mode. Unicode is a 16-bit character set that provides unique code points for all characters in various character sets. You can use the Editor in Unicode mode to represent characters from languages your terminal does not support. Unicode mode represents such undisplayable characters by a 4-digit number. You can edit records containing such characters by using up-arrow mode with Unicode enabled. You enable the Unicode mode by entering ^X.

Note: You must have NLS mode switched on to use the Unicode-enabled up-arrow mode.

In Unicode mode, non-ASCII characters are represented by their Unicode 4-digit hexadecimal numbers preceded by ^. But note that the UniVerse system delimiters, null values, and the up-arrow character itself, are always represented by their 3-digit decimal numbers. For a table of these values, see the ^ command.

Exiting the editor

When you edit a record, the Editor copies the current record into a temporary buffer. Your changes affect only the copy in the editing buffer; the original record remains unchanged. When you finish editing the record, you can save the changes. The FILE (or FI) command replaces the original record with the edited record. You can also write the changes to a different record in the file or even to a record in another file.

If you decide not to save the changes, use the EX, QUIT, or Q command. When you enter one of these commands, the Editor does not save the edited copy of the current record, and the original record remains as it was before editing. A message asks if you want to discard the changes. If you answer N , you have another chance to save the record. If you answer Y , then EX, QUIT, or Q empties the buffer without saving your changes.

After executing the FILE or QUIT command, the buffer is empty. What happens next depends on how you invoked the Editor.

If you specified only one record in the ED command, you return to the system prompt immediately after filing or exiting the record.

If you specified more than one record (as in ED *filename* *), or if you used a select list, you can use the Next Record (N), QUIT (Q), Exit (EX), or FILE (FI) command to finish editing the current record and go on to the next one. When you use N, QUIT, or EX to go on to the next record, the Editor warns you if the record has been changed and asks you if you want to discard the changes. When all selected records have been processed, the Editor prompts for another record ID. Enter another record ID, or press Enter to exit the Editor.

You cannot use EX, QUIT, or N to exit both the current record and the Editor when a select list is active. Use the X command to discard an active select list and exit the Editor without saving changes to the current record. This returns you to the system prompt.

UNIX editors

The UniVerse Editor is one of several editors available to you on UNIX platforms. You can also use the UNIX full-screen editor *vi* on any record in a UniVerse file. UniVerse provides two commands for invoking the *vi* editor: VIand UV.VI.

Syntax

Use VI for editing records in type 1 and type 19 files. Use UV.VI to edit records in a hashed UniVerse file. The syntax of the VI command is as follows:

VI [*pathname*]

pathname is the relative or absolute UNIX path of the record you want to edit.

For example, the program PAYROLL is stored as a record in the type 1 file BP. To edit it with the *vi* editor, enter the following:

```
>VI BP/PAYROLL
```

The simple syntax of UV.VI is as follows:

UV.VI *filename records*

You can also use the UV.VI command to edit records in a type 1 or type 19 file if you prefer to use UniVerse file naming conventions rather than UNIX paths—especially if the file or record names are longer than 14 characters.

For details about how to use the *vi* editor, see the UNIX documentation supplied with your system, or consult one of the books on the subject.

Chapter 2: Editor commands

This chapter describes every command available to users of the UniVerse Editor.

The commands are in alphabetical order with each command starting on a new page. The command name and a brief definition are followed by an explanation of its use and examples.

<Enter>

Pressing the Enter key at the Editor prompt displays one line at a time.

Syntax

<**Enter**>

Description

The Editor moves the current line pointer to the next line and displays it. When you press the Enter key at the bottom of a record, the Editor moves the current line pointer to the top of the record.

Example

This example shows how to use Enter to display one line at a time:

```
Top.
----: <Enter>
0001: JAMES
----: <Enter>
0002: SANDERS
----: <Enter>
                .
                .
        .
0008: 1960
Bottom at line 8.
----: <Enter>
Top.
----:
```

line#

Enter a line number at the Editor prompt to move the current line pointer to another line.

Syntax

[**+** | **-**] *line#*

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
line#	Moves the line pointer forward or backward to the line number specified.
+line#	Moves the line pointer forward the specified number of lines.
–line#	Moves the line pointer backward the specified number of lines.

Examples

This example shows how to change position from the top of the record to line 4:

```
Top.
----: 4
0004: PIEDMONT
----:
```

Line 4 appears, followed by the Editor prompt. The next command that you issue affects line 4 as the current line.

The next example shows two ways to move to a new line:

```
Top.
----: 2
0002: SANDERS
----: +2
0004: PIEDMONT
----:
```

When you enter `2` , the position changes to line number 2. When you enter `+2` , the current line moves forward two lines to line 4.

^

Use the ^ (Up-arrow) command to enable or disable up-arrow display mode in the Editor. Up-arrow display mode lets you enter and display nonprinting characters such as UniVerse system delimiters or foreign characters that your terminal does not support.

Syntax

^ [**X**]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
X	Switches on Unicode mode. To use Unicode mode, NLS mode must be on; otherwise, ^X returns an error.

Description

When you enter ^ without the `X` , you switch up-arrow display mode on or off. In up-arrow mode, system delimiters and null values are represented by their ASCII decimal codes. Use the `?` (Inquire) command to find out if up-arrow mode is on or off.

Unicode mode lets you see non-ASCII characters represented by their Unicode 4-digit hexadecimal numbers, except for system delimiters and null values. These special cases are always represented by their ASCII decimal code whether Unicode mode is switched off or on. To switch off Unicode mode, enter ^ .

You can input a system delimiter by entering its ASCII decimal code preceded by ^, as follows:

ASCII Code	Meaning	Use
^255	Item mark	Separates records in a file.
^254	Field mark	Separates fields in a record.
^253	Value mark	Separates values in a field.
^252	Subvalue mark	Separates subvalues of values in a multivalued field.
^251	Text mark	Denotes breaks in text for strings formatted with a text format option.
^128	Null value	Indicates an unknown value.

Examples

In the following example, field 8 contains three values. First they are shown with up-arrow mode off; the value marks appear as the symbol V . Then the ^ command turns up-arrow mode on; the value marks appear as ASCII character code ^253.

```
0008: XT100 V XL150 V XC250
 ----: ^
Up arrow display mode          = enabled
0008: XT100^253XL150^253XC250
```

You can use the Editor to enter special characters whether or not up-arrow mode is enabled. To insert special characters into a record, enter the ASCII code equivalents.

In the next example, a fourth value is appended to the multivalued field on line 8. The up-arrow mode is off.

```
0008: XT100  V XL150 V XC250
 ----: A ^253XT300
0008: XT100 V XL150 V XC250 V XT300
```

Notice that when the line is displayed, the value mark is displayed as a V , not as ^253. The symbol V represents a number of different ways an individual terminal can display the value mark.

In the next example, Unicode mode is switched on. The name Andrea is changed to André on a terminal that does not support the accented character é. The é is entered using its Unicode 4-digit hexadecimal value (x00E9) preceded by ^.

```
Top
----: ^X
Up arrow display mode         = enabled+Unicode
----: P
0001: Heitmann
0002: Andrea
0003: BA^253PhD
Bottom at line 3.
----: 2
0002: Andrea
----: C/ea/^x00E9/
0002: Andr^x00E9
```

```
----:
```

Note that you can enter Unicode values, for example, ^x00E9, whether Unicode mode is switched off or on.

In this case, if your system supports the ISO8859-1 character set, you could also enter this character (é) with NLS switched off, by entering `^233` (its ASCII value). But if you want to enter the formula 2pr in a record, you can do so only with NLS enabled. For example:

```
Top
----: ^X
Up arrow display mode        = enabled+Unicode
----: P
0001: 2r
Bottom at line 1.
----: C/2/2^x03C0/
0001: 2^x03C0r
----:
```

Enter the character p using its Unicode 4-digit hexadecimal value (x03C0) preceded by ^. Even if Unicode mode is off, you see non-ASCII characters that are undisplayable on your current terminal as the Unicode 4-digit hexadecimal value preceded by ^x. You see displayable non-ASCII characters as the number preceded by ^.

?

Use the `?` (Inquire) command to display a status report of the current record.

The `?` displays the name of the account, file and current record, the current line number, the setting of the up-arrow mode, the block lines that are set up, the prestored command status, and status of the change command that `OOPS` can restore.

Notice that a command ending with a question mark (`?`) is placed on the Editor command stack but is not executed.

Syntax

?

Examples

This example shows the information displayed by the Inquire command.

```
0003= 131 CRESTVIEW
----: ?
Account name            = phil
File name               = DISTRIBUTORS
Record name             = ER3120
Current line number     = 3
Up-arrow display mode   = disabled
Command execution ABORT = disabled.
BLOCK operation verification = enabled.
No BLOCK currently defined.
No Pre-stored command has been executed this session.
No changes, or the OOPS command has already been executed.
```

The next example shows that a block is defined:

```
----: ?
```

```
Account name             = phil
File name               = MEMBERS
Record name             = 1145
Current line number     = 7
Up-arrow display mode   =   disabled
Command execution ABORT = disabled.
BLOCK operation verification = enabled.
BLOCK is from line 3 through 5.
No changes, or the OOPS command has already been executed.
```

<>

Use the < (Begin Block) command to define the beginning of a block and the> (End Block) command to define the end of a block.

Syntax

<

>

Description

To set a block, move to the line you want as a boundary and use the Begin Block (<) or End Block (>) command.

To cancel a block setting, use either the < or the > command at the top of the record. Use the T (Top) command to move the current line pointer to the top of the record, then enter < or > .

You can use the ? (Inquire) command to see if any blocks are set.

Examples

This example shows the definition of a block beginning at line 3 and ending at line 5:

```
Top.
----: P
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
0007: ACME SALES
0008: 1960
Bottom at line 8.
----: 3
0003: 131 CRESTVIEW
----: <
Block FROM set to line 3.
----: 5
0005: NY
----: >
Block THROUGH set to line 5.
```

You can also define a single-line block. To do this, use the Begin Block and End Block commands on the same line, as in the following example:

```
----: P
```

```
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
0007: ACME SALES
0008: 1960
Bottom at line 8.
----: 3
0003: 131 CRESTVIEW
----: <>
Block FROM set to line 3.
Block THROUGH set to line 3.
```

The next example shows how to cancel a block setting:

```
0005: 131 CRESTVIEW
0006: PIEDMONT
0007: NY
0008: 1960
----: T
Top.
----: <
```

The current BLOCK pointers have been cleared.

A

Use the `A` (Append) command to append a string of characters to the end of the current line.

To append the same string to several lines, move to a new line and enter A. The Editor uses the string from the last Append command issued. Repeat as often as you want.

Syntax

A *string*

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
string	The character to be appended to the current line.

Example

This example shows how to use the Append command. Note that there are two spaces between the A and REPRESENTATIVE, so that the resulting line has a space between SALES and REPRESENTATIVE.

```
----: 7
0007: ACME SALES
----: A REPRESENTATIVE
0007: ACME SALES REPRESENTATIVE
----:
```

In the following example, a fourth value is appended to the multivalued field on line 8. The up-arrow mode is off.

```
0008: XT100 V XL150 V XC250
----: A ^253XT300
0008: XT100 V XL150 V XC250 V XT300
```

Notice that when the line is displayed, the value mark is displayed as a V , not as ^253. The symbol V represents a number of different ways an individual terminal can display the value mark.

ABORT

Use the ABORT command to enable and disable a pause/abort capability in the Editor.

Syntax

ABORT

Description

ABORT works like a toggle switch. By default, pause/abort is disabled when you first invoke the Editor. Use the ABORT command to enable the pause/abort capability, and use the ABORT command again to disable it.

Once ABORT is enabled, you can press any key on the keyboard (except Q) to pause the Editor command currently being executed, and then press any key again (except Q) to continue execution of the command. If you press Q (QUIT), the command being executed aborts.

B

Use the B (Bottom) command to position the current line pointer at the bottom of the current record.

Syntax

B

Description

This command is frequently used to move to the end of a record to insert lines.

Example

This example shows how to move to the top and bottom of a record:

```
0004: PIEDMONT
----: T
Top.
----: 1
0001: JAMES
----: B
0005: NY
Bottom at line 5.
----:
```

B string

Use the Break command to divide the current line into two separate lines.

Syntax

B [*string*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
string	The point after which the new line begins. The new line follows the current line in the record and line numbers adjust accordingly. Include a space between B and *string*.

Example

The following example shows two line-break commands. The first `B` command breaks the line after the first occurrence of E. The P2 prints two lines to display the characters up to and including E that form the current line. The second command specifies a string of two letters, dividing the line after ES.

```
----: 7
0007: ACME SALES REPRESENTATIVE
----: B E
0007: ACME
----: P2
0007: ACME
0008:  SALES REPRESENTATIVE
----: B ES
0008:   SALES
----: P2
0008:   SALES
0009:  REPRESENTATIVE
```

BLOCK

Use the `BLOCK` command to enable and disable verification of the starting and ending line numbers of a block during an editing session.

Syntax

BLOCK

Description

`BLOCK` works like a toggle switch during a copy, move, delete, or other editing operation. If `BLOCK` is disabled, there is no prompt to verify the block. If `BLOCK` is enabled, you must respond to the prompt before you can continue editing.

Example

This example shows the effect of the BLOCK command:

```
>ED RECORDS
REC95 lines long.
 ----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
Bottom at line 5.
----: 1
0001: LINE1
----: <
Block FROM set to line 1.
----: B
0005: LINE5
Bottom at line 5.
----: >
Block THROUGH set to line 5.
Bottom at line 5.
----: COPY
BLOCK from 1 through 5. OK (Y)  Y
0006: LINE1
----: B
0010: LINE5
Bottom at line 10.
----: BLOCK
BLOCK operation verification = disabled.
0010: LINE5
Bottom at line 10.
----: COPY
BLOCK from 1 through 5.
0011: LINE1
----: P
0011: LINE1
0012: LINE2
0013: LINE3
0014: LINE4
0015: LINE5
Bottom at line 15.
----: FI
"REC9" filed in file "RECORDS".
```

C

Use the C (Change) command to replace an old string of characters with a new string.

Syntax

C/*string* [/ [*new.string*]] [/ [**G**] [**B** | *lines*]]

Parameters

The following table describes each parameter of the syntax.

Parameters	Description
/	Delimits elements of the syntax. Use a character that is not in the data you are changing. See the following table for the allowed delimiters.
string	The text you want to change.
new.string	The text that is to replace the old string.
G	(Global) Changes every instance of string in the specified line.
B	(Block) Specifies that the change is to occur within the current block.
line	The number of lines after the current line (inclusive) in which to search for *string*.

!	@	#	$	%	&	*
/	\	:	=	+	-	
(	)	{	}	[	]	
‘	‘	.	|	“	,	

To change a string in the current line, use the following syntax:

C/*string*/*new.string*

To modify a specified number of lines or all the lines that contain string, use the following syntax:

C/*string*/*new.string*/[**G**] *lines*

If you do not specify lines, the search is performed only on the current line. If you do not specify G (for global), only the first occurrence of string on each line is changed to new.string. If you include G, every occurrence on a line is changed. This use of Change is helpful when modifying UniVerse BASIC programs.

To delete a string in more than one line, use the following syntax:

C/*string*//[**G**] *lines*

Use the Change command with the B option to change or replace a string in the current block. Define the block before you use the B option.

C/*string*/*new.string*/**B**

Use the B and G options together to change or replace every occurrence of a string in the block:

C/*string*/*new.string*/[**GB** | **BG**]

Description

The Change command can modify or delete all or part of a line, modify a specified number of lines containing a string, or modify all the lines containing a string.

Example

Using a delimiter other than a slash

The following example uses quotation marks (") as a delimiter character because the line contains a slash:

```
0011: 4/4
----: C"/4"/2"
0011: 4/2
```

Modifying one line

The following example shows how to use the Change command to modify the current line. Change searches the current line to find the string 60 and replaces it with the new string 82.

```
----: 8
0008: 1960
----: C/60/82
0008: 1982
```

Changing a string in more than one line

The following example shows six lines of text with multiple occurrences of CHAR:

```
----: P
0001: PRINT "ENTER A CHAR":
0002: INPUT CHAR1
0003: PRINT "ENTER ANOTHER CHAR":
0004: INPUT CHAR2
0005: CALL ADDEMUP(CHAR1,CHAR2,CHAR3)
0006: PRINT CHAR1:"PLUS":CHAR2:"IS":CHAR3
.
.
.
Bottom at line 21.
----:
```

This program has 21 lines, but suppose you want to change CHAR to NUMBER in the first 6 lines. Enter the following Change command at the top of the record:

```
Top.
----: C/CHAR/NUMBER/G6
0001: PRINT "ENTER A NUMBER":
0002: INPUT NUMBER1
0003: PRINT "ENTER ANOTHER NUMBER":
0004: INPUT NUMBER2
0005: CALL ADDEMUP(NUMBER1,NUMBER2,NUMBER3)
0006: PRINT NUMBER1:"PLUS":NUMBER2:"IS":NUMBER3
At line 6
----:
```

If you do not specify the number of lines, the command changes only the first line. The G ensures that every occurrence of CHAR on a line is changed, as in lines 5 and 6 where CHAR occurs three times.

Changing a string in a block

The following example shows how to change a string in a block:

```
----: T
Top.
----: P
0001: JAMES
0002: SANDERS
0003: SNOWMOBILES
0004: ACME SALES
0005: 131 CRESTVIEW
0006: PIEDMONT
0007: NY
0008: 1960
Bottom at line 8.
```

```
----: C/ie/ei/BG
BLOCK from 5 through 7. OK (Y) Y
0005: 131 CRESTVEIW
0006: PEIDMONT
----: T
Top.
----: P
0001: JAMES
0002: SANDERS
0003: SNOWMOBILES
0004: ACME SALES
0005: 131 CRESTVEIW
0006: PEIDMONT
0007: NY
0008: 1960
Bottom at line 8.
```

CAT

Use the `CAT` (Concatenate) command to join the current line and the next line, making a single line.

Syntax

CAT [*string*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
string	Text to be inserted after the current line and before the next line.

Description

You can specify a string to be inserted after the current line and before the next line.

Example

This example concatenates lines 8 and 9 and adds the new text, AND SERVICE:

```
0008: SALES
0009: REPRESENTATIVE
----: 8
0008: SALES
----: CAT AND SERVICE
0008: SALES AND SERVICE REPRESENTATIVE
----: 9
0009: 1960
```

The example shows how lines 8 and 9 look before they are joined. Line 8 becomes the current line.

Notice that spaces are included both before and after the string AND SERVICE. These spaces separate the new text from the old.

The new line 8 contains the result of the Concatenate command. The new line 9 now contains 1960 (previously on the old line 10).

COL

Use the COL (Column) command to display relative column positions on the screen.

Syntax

COL

Description

The COL command lets you position the line pointer within the line, which can be useful when entering BASIC programs.

Example

In an 80-column display, the Editor uses the first six columns to display a four-digit line number, a colon, and a space. COL marks the location of the remaining 74 column positions.

```
1----: COL
               1 2 3 4 5 6 7
12345678901234567890123456789012345678901234567890123456789012345678901234
```

COPY

Use the COPY command to copy a block of text to another location in the record.

Syntax

COPY

Description

You must use the BLOCK command to define a block before you copy it. The original block remains in its location so that there are two copies of the block in the record. Then move to the line where you want to put the copied block and use the COPY command.

The Editor displays the line numbers of the block and asks you to verify that you want to copy the block. The first line of the copy becomes the current line. Note that the COPY command cannot enclose the destination line.

If you try to copy a block before the block is set up, you get the message:

BLOCK not set up.

Example

The following example shows the eight lines of the record before using the COPY command:

```
1 ----: P
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
```

```
0005: NY
0006: SNOWMOBILES
0007: ACME SALES
0008: 1960
Bottom at line 8.
```

`PB` prints the current block:

```
1 ----: PB
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0008: 1960
Bottom at line 8.
```

Now the pointer is put at line 6 and the `COPY` command is used. The Editor prompts for block verification.

```
1 ----: 6
0006: SNOWMOBILES
----: COPY
BLOCK from 3 through 5. OK (Y) Y
0007: 131 CRESTVIEW
----: T
```

Lines 7 through 9 show the copied block:

```
1 Top.
----: P
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
0007: 131 CRESTVIEW
0008: PIEDMONT
0009: NY
0010: ACME SALES
0011: 1960
Bottom at line 11.
```

D and DE

Use the `D` or `DE` (Delete Lines) command to remove one or more lines from the current record.

Syntax

D[*lines*]

DE[*lines*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
lines	The number of lines to delete, beginning with the current line. Do not include a space between the `D` or `DE` and *lines*.

Example

In the following example, the `DE` command removes the current line, line 5, from the record. DE4 removes four lines from the top of the record.

>**ED RECORDS REC6**8 lines long.

```
----: P0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
0007: ACME SALES
 0008: 1960
Bottom at line 8.
----: 5
0005: NY
----: DE
----: T
Top.
----: P
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: SNOWMOBILES
0006: ACME SALES
0007: 1960
Bottom at line 7.
----: T
Top.
----: DE4
----: P
0001: SNOWMOBILES
0002: ACME SALES
0003: 1960
Bottom at line 3.
----: FI
"REC6" filed in file "RECORDS".
```

DELETE

Use the `DELETE` command to remove the record you are editing from the file.

Syntax

`DELETE`

Description

You cannot retrieve a deleted record. The Editor verifies that you want to delete the record before actually doing so. If you answer `Y` , the record is deleted. The Editor prompts you for another record or

returns to the UniVerse prompt, depending on how you invoked the Editor. If you answer N , the record is not deleted and you remain in the Editor.

Note: DELETE deletes the entire current record. D or DE deletes only the current line.

The FD command is a synonym for the DELETE command.

DROP

Use the DROP command to delete a block from a record.

Syntax

DROP

Description

The Editor asks you to verify that you want to delete the block.

Example

The following example shows the block, the DROP command, and the record after the block has been deleted:

```
0004: ACME SALES
----: PB
0005: 131 CRESTVIEW
0006: PIEDMONT
0007: NY

0004: ACME SALES
----: DROP
BLOCK from 5 through 7. OK (Y) Y
0004: ACME SALES
----: T
Top.
----: P
0001: JAMES
0002: SANDERS
0003: SNOWMOBILES
0004: ACME SALES
0005: 1960
Bottom at line 5.
```

DUP

Use the DUP (Duplicate) command to make one or more copies of the current line.

Syntax

DUP [*n*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
n	The number of times that the line is to be duplicated.

Description

If you do not specify a number *n*, one copy of the line is made. Each copy is placed on a line following the current line. The first copied line becomes the current line.

The Duplicate command is often used with the SEQ, on page 62 (Sequence) command to define an array in a BASIC program.

Example

This example shows how to use `DUP` to make two copies of the current line. `DUP` makes two copies of line 4 on lines 5 and 6. Line 5 becomes the current line.

```
----: 4
0004: PIEDMONT
----: DUP2
0005: PIEDMONT
----: T
Top.
----: P
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: PIEDMONT
0006: PIEDMONT
0007: NY
0008: SNOWMOBILES
0009: ACME SALES
0010: 1960
```

EX

Use the `EX` (Exit) command to exit the current record without saving changes.

Syntax

EX

Description

If a select list is active, the next record in the list is displayed. Upon executing the `EX` command, if there are any unsaved changes, the Editor prompts you as follows:

```
***** Record changed. OK to quit (Y) ?
```

Enter `Y` to exit without saving the current version of the record. Enter `N` to continue editing the record.

The `Q` and `QUIT` commands are synonyms for the `EX` command.

F

Use the `F` (Find) command to search for a string by its column location in a line.

Syntax

F [[*col#*]*string*]

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
col#	The column number with which to start searching for *string*. Do not include a space between the F and the column number. If you do not specify a column number, Find searches the first column.
string	The string of characters you want to find. Include a space between the F and the string, or between the column number and the string. If you do not specify a string and a Find command was executed earlier, the search for the previous string is repeated.

Description

The Find command moves to the next line in the current record that has the specified string starting in the specified column. If the string is not found, the line pointer moves to the next line.

Example

In the following example, the Find command finds the next line with C in the fifth column by specifying the column number. Then it finds the next line with N in the first column (the default, when the column is not specified).

```
Top.
----: F5 C
0003: 131 CRESTVIEW
----: F N
0005: NY
```

FANCY.FORMAT

Use the FANCY.FORMAT command to format BASIC source statements into a logical block by indenting, putting labels on their own line, deleting leading spaces before comments, and putting a space on each side of any comparison and assignment operators.

Syntax

FANCY.FORMAT

Description

If a line begins with a number, FANCY.FORMAT puts a colon (:) after the number.

Example

In this example, a UniVerse BASIC program is formatted with FANCY.FORMAT before it is filed:

```
>ED RECORDS REC
```

```
7 lines long.
----: P
0001:   FANCY.FORMAT Example
0002: 10*
0003: FOR X=1 TO 10
0004: PRINT X
0005: NEXT X
0006: STOP
0007: END
Bottom at line 7.
----: FANCY.FORMAT
----: P
0001:          FANCY.FORMAT Example
0002: 10:
0003:
0004:                FOR X = 1 TO 10
0005:                PRINT X
0006:             NEXT X
0007:             STOP
0008:         END
Bottom at line 8.
 ----: FI
```

FD

The FD command is a synonym for the DELETEcommand.

FI and FILE

Use the FI or FILE command to exit the current record, saving the changes made to the record.

Syntax

FI [[*filename*] *record*]

FILE [[*filename*] *record*]

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
filename	A file other than the current one. When you specify a new file name, the file must be an existing UniVerse file. You must also enter a record name, even if the name is to be the same as the current record.
record	The name of a record. If you specify only a record name, the Editor saves it in the current file. The Editor tells you if the record already exists. To overwrite an existing record, answer Y to the Editor prompt. If you answer N , the File command is not executed.

Description

If you do not specify *filename*or *record*, FILE saves the current record in the current file. The Editor prompts you for another record or returns to the UniVerse prompt, depending on how you entered the

Editor. If a select list is active, or if you specified more than one record when you entered the Editor, FILE saves the current record and calls the next record.

Example

The following example shows how to file a record and return to UniVerse:

```
>ED DISTRIBUTORS
Record name = ER3120
8 lines long.
----: P
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
0007: ACME SALES
0008: 1960
Bottom at line 8.
----: FI"
ER3120" filed in file "DISTRIBUTORS".
File name         = DISTRIBUTORS
Record name = <Return>
>
```

Instead of pressing Enter to return to the UniVerse prompt, you could enter another record name and continue editing.

FORMAT

Use the FORMAT command to indent BASIC source statements so that the program is easier to read.

Syntax

FORMAT

Description

If a line begins with a number, FORMAT puts a colon (:) after the number. FORMAT does not put blank spaces on empty lines.

Example

In the following example, a BASIC program is formatted with the FORMAT command before it is filed:

```
>ED RECORDS REC
7 lines long.
----: P
0001: *** THIS IS AN EXAMPLE OF THE FORMAT COMMAND ***
0002: 10*
0003: FOR X=1 TO 10
0004: PRINT X
0005: NEXT X
0006: STOP
0007: END
Bottom at line 7.
----: FORMAT
```

```
----: P
0001: *** THIS IS AN EXAMPLE OF THE FORMAT COMMAND ***
0002: 10:*
0003:         FOR X=1 TO 10
0004:             PRINT X
0005:         NEXT X
0006:         STOP
0007:       END
Bottom at line 7.
 ----: F
```

G

Use the G (Go to) command to specify a new current line.

Syntax

[**G**] [*line#*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
line#	The number of the line to which you want to move the line pointer. Do not include a space between G and *line#*. If you do not specify *line#*, the Go to command moves to the next line.

G< and G>

Use G< to go to the first line in a block. Use G> to go to the last line in a block.

Syntax

G<

G>

Description

The destination line becomes the current line.

Example

This example shows how to use the G< command. The block is set from line 3 through line 5.

```
----: P
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
```

```
0007: ACME SALES
0008: 1960
Bottom at line 8.
----: G<
0003: 131 CRESTVIEW
```

HELP

Use the HELP command to list the Editor commands and their syntax.

Syntax

HELP

HELP *string*

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
string	Can be one or more characters that identify the commands you want to display. If *string* is a single letter, all the Editor commands that begin with that letter are displayed. If you enter HELP with no argument, a prompt appears. Press Enter to display descriptions of all Editor commands.

Description

If you enter HELP followed by *string*, HELP displays commands whose descriptions contain that string. This is useful when you know the function but are unsure of the specific command used to perform it.

Examples

Entering HELP S displays the following information:

```
----: HELP S
SAVE                - SAVE (FILE) a copy of this record under the original name.
SAVE name       - SAVE (FILE) a copy of this record under the specified 'name'.
SAVE f name         - SAVE (FILE) a copy of this record as record 'name' in file
'f'.
SEQ///          - Generate SEQUENTIAL numbers. Formats permitted are:
                        SEQ/from/start/# SEQ/from/start/#/inc
    where       / - is any delimiter character.
            from    - is the optional character string to replace.
        start       - is the starting sequential number.
            #           - is the number of lines to SEQUENCE, or the letter 'B'
                            (sequence in defined BLOCK).
        inc - is the optional increment (default is one).
SIZE            - Display the SIZE of this record (# of LINES/FIELDS, # of
BYTES).
SPOOL           - SPOOL entire record to the PRINTER.
SPOOL#          - SPOOL '#' lines to the PRINTER.
SPOOLHELP   - SPOOL the HELP listing to the default PRINTER.
STAMP           - INSERT a 'last modified' stamp into the record, which
                    begins with a '*' (for BASIC 'comment'), and contains the
```

```
                        account name, LOGIN name (if different from account name),
                        date and time. Used to mark when record was last changed.
```

Entering `HELP EXIT` displays the Editor commands whose descriptions contain EXIT, as follows:

```
----: HELP EXIT
EX      - EXIT the editor (same as QUIT).
Q       - QUIT - EXIT the editor.
QUIT    - QUIT - EXIT the editor.
X       - EXIT (QUIT) from the editor and abandon an active SELECT list.
```

I

Use the `I` (Insert) command to insert one or more new lines of text after the current line.

Syntax

I [*text*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
text	The text you are inserting. If you do not specify *text*, you are prompted to input lines after the current line.

Description

If Insert is the first command given when you open a record, a new line is inserted at the top of the record. Enter text followed by Enter in response to the line number prompt to insert the new line.

While you are in Insert mode, the Editor prompt changes to *nnnn*= where *nnnn* is the number of the line you are about to enter.

To enter a field mark, enter ^254 or press Ctrl-^. To enter a value mark, enter ^253 or press Ctrl-]. To enter a subvalue mark, enter ^252 or press Ctrl-|. To enter a text mark, enter ^255 or press Ctrl-T. To enter a null value, enter ^128 or press Ctrl-N.

Example

In the following example, new lines are inserted at the top of the record. At the 0009= prompt, Enter ends the insertion. A message indicates that the bottom of the record is line 8, followed by the Editor prompt. As you enter new lines in an existing record, the Editor automatically renumbers lines.

```
New record.
----: I
0001= JAMES
0002= SANDERS
0003= 131 CRESTVIEW
0004= PIEDMONT
0005= NY
0006= SNOWMOBILES
0007= ACME SALES
0008= 1960
0009= <Return>
Bottom at line 8.
```

```
----:
```

IB

Use the `IB` (Insert Before) command to insert one or more lines of text into the record beginning at the line just before the current line.

Syntax

IB [*text*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
text	The text you are inserting. If you do not specify *text*, you are prompted to input lines after the current line.

Description

`IB` is like the `I` (Insert) command in that it accepts text on the command line, or puts you in input mode if no text appears on the `IB` command line.

Note that `IB` followed by a single space puts you in input mode. It does not insert a blank line before the current line.

Examples

In the first example, one line of text is inserted. In the second example, two lines are inserted.

```
>ED RECORDS REC4
10 lines long.
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
0006: LINE6
0007: LINE7
0008: LINE8
0009: LINE9
0010: LINE10
Bottom at line 10.
----: 5
0005: LINE5
----: IB ADD THIS LINE BEFORE LINE 5
0005: ADD THIS LINE BEFORE LINE
----: T
Top
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: ADD THIS LINE BEFORE LINE 5
0006: LINE5
```

```
0007: LINE6
0008: LINE7
0009: LINE8
0010: LINE9
0011: LINE1
Bottom at line 11
----: FI
"REC4" filed in file "RECORDS".
>ED RECORDS REC5
5 lines long.
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
Bottom at line 5.
----: 3
0003: LINE
----: IB
0003= THIS IS THE FIRST LINE ADDED BEFORE LINE3
0004= THIS IS THE SECOND LINE ADDED BEFORE LINE3
0005=
 ----:
Top.
----: P
0001: LINE1
0002: LINE2
0003: THIS IS THE FIRST LINE ADDED BEFORE LINE3
0004: THIS IS THE SECOND LINE ADDED BEFORE LINE3
0005: LINE3
0006: LINE4
0007: LINE5
Bottom at line 7.
----: FI
"REC5" filed in file "RECORDS".
```

L

Use the `L` (List) command to display a specified number of lines in the current record.

Syntax

L [*lines*]

Parameter

Parameter	Description
lines	The number of lines you want to display. Do not include a space between the `L` and the number of lines.

Description

The `L` command is like the `P` (Print) command, but you must specify the number of lines to be displayed.

If you do not specify a number, the Editor assumes that L is a Locate command. If no previous Locate command was used, L moves the current line pointer to the next line. If a previous Locate command was used, L moves to the next line containing the string specified by the Locate command.

Example

This example shows how the L command displays the number of lines specified. Line 4 is the current line.

```
----: L3
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
```

The L command changes the current line. Now the current line in the example is line 6.

L string or Locate

Use the L (Locate) command to search for the next line that contains a specific string of characters.

Syntax

L [*string*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
string	One or more characters. Include a space between the L and the string.

Description

If you use L without a string and you have not executed a Locate command earlier in the session, the line pointer moves to the next line. If you have executed a Locate command and use L without a string, the Editor searches for the string from the previous Locate command. A search stops at the bottom of the record. Use another L or the T (Top) command to go back to the top of the record to continue a search.

Examples

This example shows the L command without a string. L moves ahead one line.

```
0001: JAMES
----: L
0002: SANDERS
----:
```

These examples show L with a string, followed by L without a string. The second L repeats the search. The L command changes the current line. Now the current line in the example is line 6.

```
0002: SANDERS
----: L MO
0004: PIEDMONT
----: L
```

```
0006: SNOWMOBILES
----:
Bottom at line 8.
```

`L` repeats the search but does not find any more lines.

LOAD

Use the `LOAD` command to copy one or more lines from another record to the current record.

Syntax

LOAD [*filename*] *record*

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
filename	The name of the file containing the record whose lines you want to load into the current record. If you do not include a file name, the lines are loaded from a record in the current file.
record	The name of the record whose lines are to be loaded into the current record.

Description

When you use a `LOAD` command, the Editor prompts for starting and ending line or field numbers.

Example

The following example loads two lines from record 1510 to the current record in the same file:

```
----: P
0001: JAMES
0002: SANDERS
0003: SNOWMOBILES
0004: ACME SALES
0005: 131 CRESTVIEW
0006: PIEDMONT
0007: NY
0008: 1960
Bottom at line 8.
----: LOAD 1510
Starting line/field number - 4
Ending      line/field number - 52 lines/fields loaded.
0009: Atkins Co.
```

M

Use the `M` (Match) command to search for a line that matches a specified pattern.

Syntax

M [*pattern*]

Parameter

The following table describes each parameter of the syntax.

Parameter	Description
pattern	A string of characters or any valid pattern used with the UniVerse BASIC MATCH operator or the RetrieVe MATCHING keyword. Include a space between the M and the pattern. Match moves to the next line that fully matches the pattern. If you do not specify *pattern*, Match uses the pattern from a previous Match command. If there is no previous Match command, Match moves the current line pointer to the next line.

The formats in the following list can be used to find patterns that match a portion of a line. The three dots that precede or follow the pattern indicate that the match is on part of the line. Specify the location of the pattern in the line, as follows:

Pattern	Description
...*pattern*	Line must end with *pattern*.
*pattern***...**	Line must begin with *pattern*.
...*pattern***...**	Line can contain the *pattern* anywhere in the line.
*pattern1***...***pattern2*	Line must begin with *pattern1* and end with *pattern2.*

Description

The search begins with the current line. Use another M command to go back to the top of the record to continue a search if you reach the bottom.

The Match command is like the Locate command, but is faster. M searches parts of a line according to the specified pattern, whereas L searches an entire line for the specified string.

Examples

These examples show how to use the Match command.

M SANDERS matches the pattern for the entire line:

```
Top.
----: M SANDERS
0002: SANDERS
```

M 131. . . matches the pattern at the beginning of the line:

```
----: M 131...
0003: 131 CRESTVIEW
```

M . . . CREST. . . matches the pattern anywhere on the line:

```
----: M ...CREST...
0003: 131 CRESTVIEW
```

M . . . 60 matches the pattern at the end of the line:

```
----: M ...60
0007: 1960
```

M S. . .S matches a line that starts and ends in S:

```
Top.
----: M S...S
0002: SANDERS
```

MOVE

Use the MOVE command to remove a block from its current location and put it in a new location.

Syntax

MOVE

Description

MOVE combines the functions of COPY and DROP.

You must define a block before you move it. Then move to the line where you want the block to be placed, and enter the MOVE command. The Editor displays the line numbers of the block and asks you to verify that you want to move the block. The first line of the block becomes the current line.

MOVE is like COPY in that the block cannot include the destination line.

Example

This example shows how to move a block. The block set at lines 3 through 5 is moved. When the record is viewed after the move, the block is set at lines 5 through 7.

```
----: P
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
0007: ACME SALES
0008: 1960
Bottom at line 8.
----: 7
0007: ACME SALES
----: MOVEBLOCK from 3 through 5. OK (Y) Y
0005: 131 CRESTVIEW
----: T
Top.

----: P
0001: JAMES
0002: SANDERS
0003: SNOWMOBILES
0004: ACME SALES
0005: 131 CRESTVIEW
0006: PIEDMONT
0007: NY
```

```
0008: 1960
Bottom at line 8.
```

N

Use the N (Next Record) command to exit the current record and display the next record specified by an active select list.

Syntax

N

Description

The N command can be used only if a select list is active.

If there are any unsaved changes, the Editor prompts you:

```
***** Record changed. OK to quit (Y) ?
```

Enter Y to exit without saving the current version of the record. Enter N to continue editing the current record. If you want to exit the current record and the Editor, canceling the select list, use the X (Exit Editor) command.

OOPS

Use the OOPS command to undo the last change made to a record.

Syntax

OOPS

Description

To restore a record, you must use OOPS before using another modify command. However, you can use position commands to change the current line without affecting OOPS.

Use the ? (Inquire) command to see the last change command that can be undone by OOPS.

You cannot undo the DELETE record command using OOPS.

Example

The following example shows how the OOPS command restores a record following a mistaken use of the B (Break) command:

```
0007: SALES REPRESENTATIVE
----: B S0007: S
----: P0007: S
0008:
0009: 1960
Bottom at line 9.
----: OOPS
Record restored to condition prior to command "B S".
0007: SALES REPRESENTATIVE
----:
```

P

Use the P (Print) command to display a specified number of lines in the current record.

Syntax

P [*lines*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
lines	Specifies how many lines to display beginning with the current line. If *lines* is not specified, the Editor supplies an argument from a Print command used earlier in the session. If there is no previous command, P displays a full screen of lines. The TERM command sets the screen size.

Description

The Print command is like the L (List) command, but you do not need to specify the number of lines to be displayed.

Examples

This example shows what happens when you use P without an argument for the first time during a terminal session. P displays all the lines in the record.

```
----: P
001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
0007: ACME SALES
0008: 1960
Bottom at line 8.
```

The following example specifies the number of lines to display. P4 displays four lines.

```
----: P4
001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
----:
```

The following example shows the result of using P without an argument, after specifying the number of lines. P repeats the previous argument (4) and displays the current line and the next three lines.

```
----: P
004: PIEDMONT
0005: NY
0006: SNOWMOBILES
0007: ACME SALES
```

```
----:
```

PB

Use the PB (Print Block) command to print the current block.

Syntax

PB

Description

If a block is not currently defined, the Editor displays the message:

```
BLOCK not set up.
???? Try "HELP".
```

Example

This example displays the current block. The current line pointer does not change.

```
0001: JAMES
----: PB
005: 131 CRESTVIEW
0006: PIEDMONT
0007: NY
001: JAMES
----:
```

PE and PERFORM

The PE and PERFORM commands are synonyms for the XEQ command.

PL

Use the PL (Print Lines) command to print the current line and the specified lines before or after the current line.

Syntax

PL [*lines*]

PL [*-lines*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
lines	The number of lines to be printed preceding or following the current line. This argument changes the page size to the value of *lines*. The changed value becomes the default during the current Editor session. If *lines* is positive, the lines after the current line are displayed. If *lines* is negative, the lines before the current line are displayed, ending with the current line.

Description

`PL` leaves the line pointer at the current line and displays a page of lines preceding or following the current line. The default size of the page is 20 lines, not including the current line. This default displays a full screen of data with the current line at the top of the screen.

For example, if you use `PL` while at line 200 and you have not changed the default page size of 20 lines, PL displays lines 200 through 220, leaving the current line set at 200.

Example

This example shows use of the `PL` command. The record contains 10 lines. Starting at line 5, PL3 displays the next three lines (6, 7, and 8). Then, again at line 5, PL–3 displays the previous three lines (2, 3, and 4).

```
>ED RECORDS REC8
10 lines long.
----: P
001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
0006: LINE6
0007: LINE7
0008: LINE8
0009: LINE9
0010: LINE10
Bottom at line 10.
----: 5
005: LINE5
----: PL
0005: LINE5
0006: LINE6
0007: LINE7
0008: LINE8
005: LINE5
----: PL-3
002: LINE2
0003: LINE3
0004: LINE4
0005: LINE50005: LINE5
----: FI
REC8" filed in file "RECORDS".
```

PO

Use the `PO` (Position) command to specify a new current line.

Syntax

PO [*line#*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
line#	The number of the line that you want to display. If you do not specify a line number, PO moves to the top of the record.

PP

Use the PP (Print Page) command to print a page of lines surrounding the current line.

Syntax

PP [*lines*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
lines	The number of lines specified for a page (excluding the current line).

Description

If you specify *lines* as an odd number, PP rounds it up to the next even number. Thus, for example, PP5 displays the current line and three lines before it and three lines after it, or six lines around the current line.

PP with no argument prints a page of lines surrounding the current line (the current line is in the middle of the page). The default for a page (screenful) is 10 lines before the current line and 10 lines after the current line, or 21 lines.

PP does not move the current line pointer.

Example

The following example shows use of PP*lines*, where the number of lines is 4:

```
>ED RECORDS REC4
10 lines long.
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
0006: LINE6
0007: LINE7
0008: LINE8
0009: LINE9
```

```
0010: LINE10
Bottom at line 10.
----: 4
0004: LINE4
----: PP4
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
0006: LINE6
0004: LINE4
```

Q and QUIT

The `Q` and `QUIT` commands are synonyms for the `EX` command.

R

Use the `R` (Replace) command to replace all or part of the entire line with a new string.

Syntax

R [*new.string*]

R/*string* [/ [*new.string*]] [/ [**G**] [**B** | *lines*]]

Parameter

The following table describes each parameter of the syntax.

Parameter	Description
/	Delimiters used to separate elements of the syntax. Delimiters must follow `R` immediately if you are changing part of the line. No delimiter is necessary if you are replacing the entire line. Use a delimiter character that is not in the data you are changing. See the following table for the allowed delimiters.
string	The string of characters you want to change.
new.string	The new string of characters that is to replace the old string or the lines. If this command has been issued earlier in an editing session, `R` without an argument repeats the previous argument.
`G`	(Global) Changes every instance of string in the specified line.
`B`	(Block) Specifies that the change is to occur within the current block.
lines	The number of lines after the current line (inclusive) in which to search for *string*.

!	@	#	$	%	&	*
/	\	:	=	+	—	
(	)	{	}	[	]	
‘	‘	.	|	“	,	

Example

The following example shows how to replace a single line in a record:

```
----: 2
0002: SANDERS
----: R BURTON
0002: BURTON
----:
```

RELEASE

Use the RELEASE command to make a record currently open in the Editor available to other users. When you edit a record, the Editor locks it so that no other user can access it. Use RELEASE if you want others to have access to the same record.

Syntax

RELEASE

Description

Do not use this command if you are changing the record when editing.

SAVE

Use the SAVE command to write changes made to the current record and continue editing the same record.

Syntax

SAVE [[*filename*] *record*]

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
filename	The file in which you want to write the record. To save the current record to another file, you must specify both *filename* and *record* even if you are not renaming the record.
record	The name of the saved record. If you specify only a record name, SAVE renames the record in the current file and keeps the original record in the file.

Description

The SAVE command writes the current record in the current file. After the record is saved, the record remains open for further editing.

If you invoke the Editor on an existing record in a distributed or part file and try to save (SAVE) it with a new record ID, you lose any edits for the session.

Example

This example shows how to use the SAVE command:

```
----: P
0001: JAMES
0002: SANDERS
0003: 131 CRESTVIEW
0004: PIEDMONT
0005: NY
0006: SNOWMOBILES
0007: ACME SALES
Bottom at line 7.
----: SAVE
"ER3120" filed in file "DISTRIBUTORS".
Bottom at line 7.
----:
```

SEQ

Use the SEQ command to generate sequential numbers within lines or at the beginning of lines in a record.

Syntax

SEQ/ [*string*] /*start*/ { *lines* | **B** } [/*incr*]

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
/	Delimits the string and starting number. You must include the first and second delimiters even if you do not specify *string*.
string	The string to replace. If you specify *string*, the sequential number replaces the original string.
start	The starting number of the sequence.
lines	The number of lines to number, starting with the current line.
B	Specifies that only the lines currently defined in the block should be numbered. You must specify either *lines* or B.
incr	The optional number to increment. If you do not include *incr*, the default is 1.

Example

This example shows how to use the SEQ and DUP commands to set up and assign values to an array of 10 elements. The hash sign (#) represents the array index. The dollar sign ($) represents the value of each element in the array.

After you define the template line for the array, you use the DUP command to make 9 copies, giving a total of 10. SEQ replaces the $ with values of from 5 through 50 in increments of 5, and replaces each occurrence of # with a value from 1 through 10.

```
0001= ARRAY(#) = $
0002=
Bottom at line 1.
----: DUP 9
0002: ARRAY(#) = $
----: P
0002: ARRAY(#) = $
0003: ARRAY(#) = $
0004: ARRAY(#) = $
0005: ARRAY(#) = $
0006: ARRAY(#) = $
0007: ARRAY(#) = $
0008: ARRAY(#) = $
0009: ARRAY(#) = $
0010: ARRAY(#) = $
Bottom at line 10.
----: PO 1
0001: ARRAY(#) = $
----: SEQ/$/5/50/5
0001: ARRAY(#) = 5
0002: ARRAY(#) = 10
0003: ARRAY(#) = 15
0004: ARRAY(#) = 20
0005: ARRAY(#) = 25
0006: ARRAY(#) = 30
0007: ARRAY(#) = 35
0008: ARRAY(#) = 40
0009: ARRAY(#) = 45
0010: ARRAY(#) = 50
Bottom at line 10.
----: T
Top.
----: SEQ/#/1/10
0001: ARRAY(1) = 5
0002: ARRAY(2) = 10
0003: ARRAY(3) = 15
0004: ARRAY(4) = 20
0005: ARRAY(5) = 25
0006: ARRAY(6) = 30
0007: ARRAY(7) = 35
0008: ARRAY(8) = 40
0009: ARRAY(9) = 45
0010: ARRAY(10) = 50
Bottom at line 10.
```

SIZE

Use the SIZE command to display the size (number of lines and bytes) of the current record.

Syntax

SIZE

Example

This example shows how to use the SIZE command:

```
----: T
Top.
----: P
0001: JAMES
0002: SANDERS
0003: SNOWMOBILES
0004: ACME SALES
0005: 131 CRESTVIEW
0006: PIEDMONT
0007: NY
0008: 1960
Bottom at line 8.
----: SIZE
Size of "ER3120" is 8 lines and 68 bytes (characters).
Bottom at line 8.
```

SPOOL

Use the SPOOL command to spool all or part of the record to the printer.

Syntax

SPOOL [*lines*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
lines	The number of lines to spool. If you do not include a number, the entire record is spooled. You must be in edit mode to use SPOOL.

Examples

These examples show both uses of SPOOL:

```
----: SPOOL
"1115" spooled to the printer.
----: SPOOL 3
Lines 1 to 3 of "1115" spooled to the printer.
```

SPOOLHELP

Use the SPOOLHELP command to spool the Editor help listing to a printer.

Syntax

SPOOLHELP

Example

SPOOLHELP produces the following message:

```
Editor HELP record has been spooled to the printer.
```

STAMP

Use the STAMP command to mark the date and time a record was last changed.

Syntax

STAMP

Description

The stamp begins with an asterisk (*) like a BASIC comment and contains the following:

- UniVerse account name
- Login name, if it differs from the UniVerse account name
- Date
- Time

Example

Here is an example of the STAMP command:

```
>ED RECORDS REC
17 lines long
----: P
0001: THIS IS A TEST
0002: LINE1
0003: LINE2
0004: LINE3
0005: LINE4
0006: LINE5
0007: THIS IS A TEST
Bottom at line 7.
----: STAMP
0008: *         Last updated by manuals (jewel) at 11:57:19 on
05/12/1995
Bottom at line 8.
----: FI
"REC1" filed in file "RECORDS".
```

T

Use the T (Top) command to position the current line pointer at the top of the record.

Syntax

T

Description

This command is frequently used to move to the beginning of a record to insert lines.

Example

This example shows how to move to the top and bottom of a record:

```
0004: PIEDMONT
----: T
Top.
----: 1
0001: JAMES
----: B
0005: NY
Bottom at line 5.
----:
```

UNLOAD

Use the UNLOAD command to copy all or part of the current record to another record.

Syntax

UNLOAD [*filename*] *record*

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
filename	The name of the file to which you are copying all or part of the current record. If you do not include *filename*, UNLOAD saves specified lines as another record in the current file.
record	The name of the record to which you are copying.

Description

When you use the UNLOAD command, the Editor prompts you for the starting and ending lines. UNLOAD copies the lines to the specified destination. If the record exists, the following message appears:

```
Record already exists. OK to overwrite (Y) ?
```

Examples

This example copies four lines from the current record to a new record in the same file:

```
----: T
Top.
----: P
0001: JAMES
0002: SANDERS
0003: SNOWMOBILES
0004: ACME SALES
0005: 131 CRESTVIEW
```

```
0006: PIEDMONT
0007: NY
0008: 1960
Bottom at line 8.
----: UNLOAD REC1115
Starting line/field number - 4
Ending   line/field number - 74 lines/fields unloaded.
Bottom at line 8.
----:
```

This example copies the whole record to another file. You can enter the ending line or any number larger than the number of lines in the record.

```
----: T
Top.
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
Bottom at line 5.
----: UNLOAD CUSTOMERS TEST
Starting line/field number - 1
Ending   line/field number - 6
Record already exists.  OK to overwrite (Y) ? Y
5 lines/fields unloaded.
Bottom at line 5.
----:
```

X

Use the X (Exit Editor) command to exit the current record and return to the command processor, discarding an active select list.

Syntax

X

Description

You can use the X command only if a select list is active.

If there are any unsaved changes, the Editor prompts you:

```
***** Record changed. OK to quit (Y) ?
```

Enter Y to exit without saving the current version of the record. Enter N to continue editing the record. If you want to exit the record and display the next record in the select list, use the N (Next Record) command.

XEQ

Use the XEQ (Execute) command to execute a UniVerse command from within the Editor.

Syntax

XEQ

Description

After the command executes, you return to the Editor.

XEQ uses the following variables to supply the current file name, record name, line number, and so on:

Variable	Description
@FILE	Current file name
@ID	Current record name
@LINE	Current line number
@FM	Field mark
@VM	Value mark
@SM	Subvalue mark

The PE and PERFORM commands are synonyms for the XEQ command.

Example

In the following example, XEQ executes a UniVerse LIST command from within the Editor:

```
Top.
----: P
001: Mr. B. Clown^2531 Center Ct.^253New York, NY 10020
0002: (918) 737-2118
0003: Home
Bottom at line 3.
----: XEQ LIST @FILE @ID
Executing the command "LIST CUSTOMERS 4450".


LIST CUSTOMERS 4450 11:13:54am  05 May 1995 PAGE   1
Cust No Bill to..................... Phone Number.
PhoneDesc
4450            Mr. B. Clown                    (918) 737-2118
Home
         1 Center Ct.
         New York, NY 10020
1 records listed.
-----------------     Returned to the EDITOR from the XEQ command.
0003: Home
Bottom at line 3.
```

.A

Use the .A (Append) command to append text to the end of a command in the Editor stack.

Syntax

.A [*line#*] *string*

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
line#	The line number in the stack to which you want to append *string*. If you do not specify *line#*, *string* is appended to sentence 1.
string	The text you want to add to the command.

Example

This example appends text to the end of line 1 in the stack:

```
02 P
01 SPOOL

Top.
----: .A HELP
01 SPOOLHELP
```

.C

Use the `.C` (Change) command to change *old* text to *new* text in the command in the Editor stack.

Syntax

.C [*line#*] */old/new*

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
line#	The line number in the stack whose text is to be changed. If you do not specify *line#*, the text in sentence 1 changes.
/	Delimits elements of the syntax.
old	The existing text you want to change.
new	The new text you want to add.

Example

This example shows how `.C` changes `SPOOLHELP` to `SPOOL` in line 5 in the stack:

```
05 SPOOLHELP

0005: LINE5
Bottom at line 5.
----: .C/HELP//
01 SPOOL
```

.D

Use the `.D` (Delete) command to delete a command from the Editor stack or to delete a record from the &ED& file or another file, if specified.

Syntax

.D*line#*

.D*record*

.D *file record*

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
line#	The line number in the stack to be deleted. If you do not specify *line#*, line 1 is deleted.
record	The name of a stored command record in the &ED& file.
file	The name of the file where the record is stored. If *file* is not specified, the record is stored in the default &ED& file.

Example

This example shows `.D` used to delete line 1 in the stack:

```
03 P
02 SPOOLHELP
01 SPOOL
0005: LINE5
Bottom at line 5.
----: .D
History #1 DELETEd.
0005: LINE5
Bottom at line 5.
----: .L
02 P
01 SPOOLHELP
0005: LINE5
Bottom at line 5.
```

.I

Use the `.I` (Insert) command to insert a new command into the Editor stack.

Syntax

.I*line#*

.I*line# string*

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
line#	The line number in the stack after which *string* is to be inserted. If you do not specify *line#*, *string* is inserted at line 1.
string	A string of characters, usually an Editor command, to be inserted. If you do not specify *string,* the Editor prompts you to enter a command at *line#.*

Example

This example shows `.I` used to insert a command into the stack. Line 3 in the stack is replaced by the new line.

```
0003: Trick Room
Bottom at line 3.
 ----: .L
03 P
02 T
01 B
0003: Trick Room
Bottom at line 3.
----: .I3 SPOOLHELP
03 SPOOLHELP
0003: Trick Room
Bottom at line 3.
----: .L
03 SPOOLHELP
02 T
01 P
0003: Trick Room
Bottom at line 3.
```

.L

Use the `.L` (List) command to list the commands in the Editor stack or the commands stored in a record in the &ED& file or *file,* if specified.

Syntax

.L*lines*

.L*record*

.L *file record*

.L*file **

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
lines	The number of lines in the stack to list. If you do not specify *lines*, lines 1 through 9 are listed.
record	The name of a stored command record in the &ED& file to list.

Parameter	Description
file	The name of the file where the record is stored. If *file* is not specified, the record is stored in the default &ED& file.
*	The names of prestored command records in *file. file* defaults to &ED&.

Example

This example shows `.L` used to list the commands in the stack:

```
----: .L
06 P
05 T
04 B
03 SPOOLHELP
02 T
01 P


0003: Trick Room
 Bottom at line 3.
 ----:
```

.R

Use the `.R` (Recall) command to recall a command in the Editor stack, or a stored command record from the &ED& file or *file* if specified.

Syntax

.R*line#*

.R *record*

.R *file record*

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
line#	The line number in the stack to recall. If you do not specify *line#*, no action is taken since the default, command 1, is already in line 1.
record	The name of a stored command record in the &ED& file.
file	The name of the file where the record is stored. If no *file* is specified, the record is stored in the default &ED& file.

Description

When you recall an Editor command, it is copied to line 1. When you recall a stored command record, the commands in the record are copied to the first *n* commands in the stack.

Example

This example shows `.R` used to recall TEST from the &ED& file:

```
02 P
01 T
0003: Trick Room
Bottom at line 3.
----: .R TEST
***** Loaded 1 command(s).
```

.S

Use the `.S` (Save) command to save Editor stack commands in the &ED& file (or another file if specified) for later execution.

Syntax

.S*line# record*

.S*line# file record*

.S*record start,end*

.S *file record start,end*

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
line#	The line to save.
record	The name of the record in the &ED& file or *file* that is to store the Editor commands.
file	The name of the file where record is to be stored for later execution. If *file* is not specified, the record is stored in the default &ED& file.
start	The number of the first Editor command to save. If you specify *start* only, Editor commands from line *start* to line 1 are saved.
end	The number of the last Editor command to save.

Examples

The following example saves the record TEST in the default file &ED&. Because no file is specified and the &ED& file does not already exist, the file is created.

```
----: .S TEST
Creating file "&ED&" as Type 1.
Creating file "D_&ED&" as Type 3, Modulo 1, Separation 2.
Added "@ID", the default record for RetrieVe, to "D_&ED&".
Saved "TEST" in file "&ED&".
```

The next example saves a list of five commands on the stack as a record EXP in the &ED& file:

```
----: .L
```

```
05 P
04 XEQ LIST @FILE
03 SPOOL
02 P
01 5 LIST VOC

0005: LINE5
Bottom at line 5.
----: .S5 EXP
Saved "EXP" in file "&ED&".

----: .L EXP
    EXP
001 EPre-stored Command saved at 15:56:40 15 May 1995
002 P
003 XEQ LIST @FILE
004 SPOOL
005 P
006 5 LIST VOC
```

The following example saves lines 13 through 4 in the command stack into the record TEST in the ALT.ED file:

```
---- .S ALT.ED TEST 13, 4
Saved "TEST" in file "ALT.ED".

0006: SPOOL
Bottom at line 6.
----:
```

Note that the following example, without the comma, saves command 1 in the record 5 4 in the STACK.S3 file:

```
----: .S STACK.S3 5 4
Saved "5 4" in file "STACK.S3".

0005: LINE5
Bottom at line 5.
```

The following example saves command 1 in the record 3 in the STACK.S3 file:

```
----: .S STACK.S3 3
Saved "3" in file "STACK.S3".
```

Using PAUSE and LOOP commands

You can include `PAUSE` and `LOOP` commands within the stored command record. Use `PAUSE` to interrupt execution of the command series. When the stored commands are executed and `PAUSE` interrupts execution, use either `.XK` to abort execution, or `.XR` to resume execution. Use `LOOP` to repeat a sequence of Editor commands in the stored record.

Syntax

The syntax for the `LOOP` command is:

LOOP [*start*] [*rep*]

Parameter

The following table describes each parameter of the syntax.

Parameter	Description
start	The starting line number of the loop.
rep	The number of times to repeat the looping sequence. The LOOP command is the ending line.

If you do not specify *start* or *rep*, the default for both is 1. Use .I to insert the LOOP command in the command stack before using .S to store the command record.

Example

This example shows a stored command record using the PAUSE and LOOP commands:

```
>ED &ED& EXQ
5 lines long.
----: P
0001: EPre-stored Command saved at 16:02:08 11 Apr 1995
0002: XEQ TIME
0003: XEQ USERS
0004: PAUSE
0005: LOOP 2 3
Bottom at line 5.
```

The following example shows the LOOP command repeating a sequence of C (Change) commands. The stored commands change the paths in selected VOC file pointers from */usr1*... to */usr2*... .

First you select the records you want to change:

```
>SELECT VOC WITH F2 LIKE /usr1...

27 record(s) selected to SELECT list #0.
>>ED VOC
SELECTed record name = "SALES".
3 lines long.
----:
```

Next, you execute the command sequence you want to save:

```
----: 2
0002: /usr1/SALES
----: C/usr1/usr2
0002: /usr2/SALES
----: 3
0003: /usr1/D_SALES
Bottom at line 3.
 ----: C/usr1/usr2
0003: /usr2/D_SALES
Bottom at line 3.
----: FI
"SALES" filed in file "VOC".

SELECTed record name = "INVENTORY".
3 lines long.
```

Now you enter the LOOP command and list the Editor stack to verify the command sequence:

```
----: LOOP 1 99
----: .L

06 2
05 C/usr1/usr2
04 3
03 C/usr1/usr2
02 FI
01 LOOP 1 99

Top.
```

The .S command saves lines 6 through 1 to the record CHANGE in the &ED& file:

```
----: .S CHANGE 6,1
            Saved "CHANGE" in file "&ED&".
```

The .X command executes the CHANGE record on the remaining records in the VOC file:

```
----: .X CHANGE
0002: /usr1/INVENTORY
0002: /usr2/INVENTORY
0003: /usr1/D_INVENTORY
0003: /usr2/D_INVENTORY
"INVENTORY" filed in file "VOC".

SELECTed record name = "PERSONNEL".
3 lines long.

0002: /usr1/PERSONNEL
0002: /usr2/PERSONNEL
0003: /usr1/D_PERSONNEL
0003: /usr2/D_PERSONNEL
"PERSONNEL" filed in file "VOC".
.
.
.
File name           = VOC
Record name = <Enter>
```

.U

Use the .U (Uppercase) command to convert a sentence to uppercase.

Syntax

.U[*line#*]

Parameter

The following table describes the parameter of the syntax.

Parameter	Description
line#	The number of the command you want to change. If you do not specify *line#*, line 1 converts.

Example

This example shows .U used to convert the command spoolhelp to uppercase:

```
0003: Trick Room
Bottom at line 3.
----: spoolhelp
HELP listing spooled to printer.
0003: Trick Room
Bottom at line 3.
----: .U
01 SPOOLHELP
0003: Trick Room
Bottom at line 3.
```

.X

Use the .X (eXecute) command to reexecute Editor commands stored in the Editor stack or in the &ED& file (or other file if specified).

Syntax

.X*line#*

.X *record*

.X *file record*

Parameters

The following table describes each parameter of the syntax.

Parameter	Description
line#	The command number to be executed. .X copies the command to line 1. If you do not specify *line#*, line 1 is reexecuted.
record	The name of the stored command record.
file	The file where the record is stored. If *file* is not specified, the record is stored in the default &ED& file.

Examples

This example shows .X used to reexecute a stack command that inserts a line:

```
>ED RECORDS RECORD
15 lines long.
----: P
0001: LINE1
0002: LINE2
0003: LINE3
0004: LINE4
0005: LINE5
```

```
Bottom at line 5.
----: T
Top.
----: I THIS IS A TEST
0001: THIS IS A TEST
----: B
0006: LINE5
Bottom at line 6.

----: .L
04 P
03 T
02 I THIS IS A TEST
01 B
0006: LINE5
Bottom at line 6.
----: .X2
02 I THIS IS A TEST
0007: THIS IS A TEST
Bottom at line 7.
```

This example shows `.X` reexecuting the prestored command `CHANGE` to change select file pointers in the VOC file from */u1* … to */u2* …:

```
>ED VOC
*SELECTed record name = "&DEVICE&".
 3 lines long.


----: P
0001: F
0002: /u1/uv/SALARIES
0003: /u1/uv/D_SALARIES
Bottom at line 3.
----: .X CHANGE
0003: /u2/uv/D_SALARIES
***** End of prestored command execution.
0003: /u2/uv/D_SALARIES
Bottom at line 3.
----: N
***** Record changed.   OK to go to next record (Y) ? Y
SELECTed record name = "PERSONNEL".
3 lines long.
----: P
0001: F
0002: /u1/uv/PERSONNEL
0003: /u1/uv/D_PERSONNEL
Bottom at line 3.
----: .X CHANGE
0003: /u2/uv/D_PERSONNEL
***** End of prestored command execution.
0003: /u2/uv/D_PERSONNEL
.
.
.
```

.XK

Use the `.XK` (Abort Execution) command after a `PAUSE` command in a stored command sequence interrupts execution.

Syntax

.XK

Example

The following example shows .XK aborting execution of the stored command record EXQ. First, .L lists the record, and .X reexecutes the stored Editor commands.

```
>ED RECORDS REC
17 lines long.
----: .L EXQ
        EXQ
001 EPre-stored Command saved at 16:02:08 05 May 1995
002 XEQ TIME
003 XEQ USERS
004 PAUSE
005 LOOP 1 3
Top.
----: .X EXQ
Executing the command "TIME".
12:48:10 05 May 1995
-----------   Returned to the EDITOR from the XEQ command.
Top.
Executing the command "USERS".
There are currently 7 users logged on the system.
-----------   Returned to the EDITOR from the XEQ command.
Top.
**** Prestored command PAUSE at line 3.
Top.
Pause.
----: .XK
----: FI
"REC1" filed in file "RECORDS".
```

.XR

Use the .XR (Resume Execution) command after a PAUSE command in a stored command sequence interrupts execution.

Syntax

.XR

Example

This example shows .XR resuming execution after a PAUSE. First, the .X command reexecutes the EXQ command series. Notice that the PAUSE command interrupts the execution of the EXQ commands. Then, .XR reexecutes the EXQ commands after the PAUSE command. Finally, .XK aborts the execution after PAUSE.

```
>ED RECORDS REC
15 lines long.
----: .L EXQ
            EXQ
001 EPre-stored Command saved at 16:02:08 05 May 1995
002 XEQ TIME
003 XEQ USERS
```

```
004 PAUSE
005 LOOP 1 3
Top.
----: .X EXQ
Executing the command "TIME".
12:15:52 05 May 1995
-----------  Returned to the EDITOR from the XEQ command.
Top.
Executing the command "USERS".
There are currently 7 users logged on the system.
-----------  Returned to the EDITOR from the XEQ command.
Top.
**** Prestored command PAUSE at line 3.
Top.
Pause.
----: .XR
Executing the command "TIME".
12:16:02 05 May 1995
-----------    Returned to the EDITOR from the XEQ command.
Top.
Executing the command "USERS".
There are currently 7 users logged on the system.
-----------   Returned to the EDITOR from the XEQ command.
Top.
**** Prestored command PAUSE at line 3.
Top.
Pause.
----: .XK
----: FI
"REC1" filed in file "RECORDS".
```

Chapter 3: Appendix A: Editor commands by function

This appendix summarizes the Editor commands described in detail in Editor commands, on page 26. The following table lists the tasks you can perform with the UniVerse Editor and the commands you use to perform them:

If you want to...	Use...
Abort the Editor	ABORT
Append text to a line.	A (Append)
Break lines.	B (Break)
Change lines.	C (Change)
Change the current line or move to a specific line.	*line#*, G (Go to), PO (Position)
Change the Editor mode.	I (Insert)
Change up-arrow mode to on (enabled) or off (disabled).	^ (Up-arrow)
Concatenate lines.	CAT (Concatenate)
Copy a block.	COPY
Copy all or part of a record to the current record.	LOAD
Copy all or part of the current record to a specified record.	UNLOAD
Delete a block of lines from a record.	DROP
Delete a string in one or more lines.	C (Change),R (Replace)
Delete one or more lines from the current record.	D and DE and DE (Delete Lines)
Delete the current record.	DELETE
Disable a pause/abort capability in the Editor.	ABORT
Display 10 lines before and 10 lines after the current line without moving the current line pointer.	PP (Print Page)
Display relative column positions on the screen.	COL (Column)
Display system delimiters and special characters.	^ (Up-arrow)
Display the current record's status.	<> (Inquire)
Display the next or previous 20 lines of the record without moving the current line pointer.	PL (Print Lines)
Display the next line.	L (List), <Enter>
Display the next record while editing multiple records.	EX (Exit), FI and FILE, N (Next Record), Q and QUIT, .X (Exit Editor)
Display the next screen of lines and move the current line pointer to the last line displayed.	P (Print)
Display the size of the current record.	SIZE
Duplicate a line.	DUP (Duplicate)
Edit UniVerse BASIC programs.	COL (Column)
Enable or disable the block verification prompt.	BLOCK

If you want to...	Use...
Enter and display nonprinting characters, such as UniVerse system delimiters or foreign characters, that your terminal does not support.	^ (Up-arrow)
Exit a record and save changes to the record.	FI and FILE
Exit the current record, discarding current changes.	EX (Exit), N (Next Record), Q and QUIT,X (Exit Editor)
Find a string in a line or set of lines.	L (List), L string or Locate
Format UniVerse BASIC programs.	FANCY.FORMAT, FORMAT
Generate sequential numbers within lines or at the beginning of lines in a record.	SEQ (Sequence)
Go to the first line of a defined block.	G<
Go to the last line of a defined block.	G>
Insert new lines.	I (Insert), IB (Insert Before)
Invoke the Editor.	ED
Pause the Editor.	ABORT
Mark blocks of text.	<> (Begin and End Block)
Mark the current date and time on the record.	STAMP
Move a block to the current line and delete it from its original location.	MOVE
Move the pointer to the top of the record.	T (Top)
Move the pointer to the bottom of the record.	B (Bottom)
Move to a specific line.	*line*#, G (Go to), PO (Position)
Print a block.	PB (Print Block)
Print a hard copy of all or part of a record.	SPOOL
Print a page of lines.	PP (Print Page)
Print a specified number of lines before and after the current line.	PL (Print Lines)
Print a specified number of lines in the current record.	P (Print)
Release the lock on the current record while you are editing it.	RELEASE
Replace lines.	R (Replace)
Run another UniVerse process while remaining in the Editor.	XEQ (Execute)
Save the current record and continue editing the same record.	SAVE
Search the record for the next line containing a specified string or pattern and then display the line.	F (Find),L (Locate), M (Match)
Store a group of commands in a file to make it easy to perform repetitive tasks.	Stack commands
Undo the last change made to a record.	OOPS