

# UED - Line-Oriented Source Editor Utility

UED is an online utility with which you can edit source text using any available terminal device type with both input and output capability. Specifically, UED can be used with hard copy type terminals such as TTY devices and 2741 terminals.

With UED, all editing operations are performed in editor work files allocated within Com-plete SD files. First, an SD file is allocated; then the library or member is read into the SD file. Next, you edit the data in the SD file, and finally use the SAVE command to write the data back to the indicated library member.

In addition to dynamic allocation of sequential or partitioned data sets, the UED utility includes these features:

- You can use any library containing source text as input or output to UED.
- For MVS, input and output data sets are restricted to the following DCB conventions:  
  
`RECFM=F or RECFM=FB`  
`DSORG=PS or DSORG=PO`
- In MVS, BLKSIZE value of any input or output library is restricted only by the size of the thread in which UED is to execute. If files with large BLKSIZE are encountered while using UED, editing may not be possible. In this situation, you will need the assistance of the system programmer responsible for Com-plete maintenance.
- In MVS, you can use any sequential data set containing source text as input or output to UED.
- You can select statements from other members and other libraries for insertion into the text you're editing.
- Editing is either statement number-oriented or tag-oriented, depending upon your selection.
- You may use a full range of editing commands using a well-defined syntax structure.
- A user-written subroutine can be used for establishing security restrictions.
- A table of library identification codes enables you to use a two-character reference instead of entering the full file name.

This chapter covers the following topics:

- Command Format
- UED Commands

---

## Command Format

UED is a fully conversational online utility program. This means that you can select one of the various functions available when you invoke UED, or any time thereafter.

The basic command format is:

**\*UED**

Once you have entered this command, conversation is established with UED and you can then enter the various command functions.

Optionally, you can use the command format:

**\*UED function**

where *function* is any of the command functions defined for UED.

**Note:**

Unlike the UEDIT utility, termination of UED requires that you understand the termination command END as described later in this chapter. Pressing the CLEAR key on a 3270-type terminal or "control/D" on a TTY-type terminal terminates UED *only* if you have not modified the SD work file in use.

## UED Commands

You can abbreviate all UED commands. In the command descriptions that follow, the minimum abbreviations allowed are indicated by an underscore.

<b>Function</b>	<b>Description</b>
<u>C</u> HANGE	Changes one or more occurrences of a given character string to another character string.
<u>C</u> OPY	Copies a designated set of statements.
<u>D</u> EFINE	Defines or resets the default characteristics assumed by UED during the editing session.
<u>D</u> ELETE	Deletes one or more statements.
<u>E</u> ND	Terminates the editing session.
<u>F</u> IND	Finds a designated character string.
<u>H</u> ALT	Halts the execution of a command or suppresses output from UED to the terminal.
<u>H</u> ELP	Displays the syntax for a given command (same as ? command).
<u>I</u> NSERT	Inserts one or more statements.
<u>L</u> IST	Lists one or more statements.
<u>M</u> ACRO	Defines the instream UED functions to be executed.
<u>M</u> OVE	Relocates one or more statements.
<u>N</u> UMBER	Resequences one or more statements.
<u>P</u> RINT	Obtains a hard copy listing of one or more statements.
<u>Q</u> UERY	Displays the current default values (set by using the DEFINE command).
<u>R</u> EAD	Reads or obtains from a library one or more statements or members.

Function	Description
<u>R</u> ECOVER	Recovers the UED work file after a system failure.
<u>R</u> EPEAT	Repeats a line or command as previously entered.
<u>R</u> ESUME	Resumes typing (after a HALT command).
<u>S</u> AVE	Saves all changes made during an editing session.
<u>S</u> UBMIT	Submits a job stream to batch for execution.
<u>T</u> AB	Lists the current tab settings.
+n	Pages forward n statements.
-n	Pages backward n statements.
++	Pages to the last statement.
--	Pages to the first statement.
/tag	Pages to the statement containing a specified value (tag), where tag is any string of characters that begins in the designated tag column.
n	Pages to or modifies the statement with sequence number n.
?	Synonym for HELP.

Note that you can request more than one command function at a time, provided that each function is separated by the command delimiter.

**Note:**

If an error results while a series of commands is executing, the command function in error is terminated and the subsequent commands are ignored.

You can dynamically change most default values used by UED, including the command delimiter (the default is a semicolon), at any time during execution of UED or at invocation of UED by using the DEFINE command.

The descriptions of the various UED commands in the following sections refer to subscripted items such as *num*, *addr*, and *st*. Note that the following definitions apply to all such references unless otherwise specifically stated:

num	Any positive integer, zero, or the word ALL, which specifies all applicable occurrences.														
addr	<p>Any of the following values:</p> <table border="0"> <tr> <td>number</td> <td>Specifies any statement sequence number. You need not supply zeros unless the sequence number is more than ten digits.</td> </tr> <tr> <td>string</td> <td>Specifies any string preceded by the designated tag character (the default is a slash).</td> </tr> <tr> <td>--</td> <td>Specifies the first statement of the file to be edited.</td> </tr> <tr> <td>++</td> <td>Specifies the last statement of the file to be edited.</td> </tr> <tr> <td>*</td> <td>Specifies the current statement.</td> </tr> </table> <p>You can also use one of the following values with +n or -n (where n is an integer) as a replacement for addr in order to establish relative addressing:</p> <table border="0"> <tr> <td>+n</td> <td>indicates forward relative addressing by n statement(s).</td> </tr> <tr> <td>-n</td> <td>indicates backward relative addressing by n statement(s).</td> </tr> </table>	number	Specifies any statement sequence number. You need not supply zeros unless the sequence number is more than ten digits.	string	Specifies any string preceded by the designated tag character (the default is a slash).	--	Specifies the first statement of the file to be edited.	++	Specifies the last statement of the file to be edited.	*	Specifies the current statement.	+n	indicates forward relative addressing by n statement(s).	-n	indicates backward relative addressing by n statement(s).
number	Specifies any statement sequence number. You need not supply zeros unless the sequence number is more than ten digits.														
string	Specifies any string preceded by the designated tag character (the default is a slash).														
--	Specifies the first statement of the file to be edited.														
++	Specifies the last statement of the file to be edited.														
*	Specifies the current statement.														
+n	indicates forward relative addressing by n statement(s).														
-n	indicates backward relative addressing by n statement(s).														
st	<p>Any string of characters.</p> <p>Delimiter UED command delimiter for strings If the character string contains one or more delimiters (for example, blanks, semicolons, commas) or is a null character string, the string must be enclosed by string delimiters (for example, single quotation marks). If the string contains a quotation mark, the default string delimiter and the quotation mark must appear twice.</p> <p>Note that the default string delimiter (single quotation marks) is overridden using the DEFINE command.</p>														

Note that several arguments in the following command descriptions are defined as *addr* arguments but their descriptions indicate that they may be *num* (numbers) as well. Such arguments must be preceded by the default number identifier # in order to ensure that the number attribute is attached.

Note also that all commands except DEFINE have positional arguments. You must replace any intermediate argument that is to be omitted for a given command by an operand delimiter (for example, comma). If you omit trailing arguments, however, a replacement operand delimiter is not required.

You can use the SAVE command to replace an existing member of a source library. If the member does not exist, a new member is created.

In MVS, you can overwrite an existing sequential file with the SAVE command.

## CHANGE Command

The CHANGE command enables you to change one or more occurrences of a given character string within a statement to another character string. You can define the number of occurrences per statement (one line) as well as the number of statements.

The command format is:

```
CHANGE st1,st2[,num][,addr1][,addr2]
```

The arguments are:

st1	Required.
	Specifies the character string to be changed.
st2	Required.
	Specifies the replacement character string.
num	Optional.
	Default: ALL
	Specifies the maximum number of occurrences per statement of the character string for which the change will occur.
	Note that the change occurs left to right, per statement, for this number of occurrences.
addr1	Optional.
	Default: * (indicates the current statement)
	Specifies the address of the first statement for which the change is to occur.
addr2	Optional.
	Default: #1
	Specifies the address of the last statement for which the change occurs or, if stated as a number (e.g., #25), the number of statements to be affected.

Note that you can only change a character string when the entire string is contained within the following ranges:

- Before the sequence columns;
- In the sequence columns;
- After the sequence columns.

## COPY Command

The COPY command enables you to copy one or more statements.

The command format is:

```
COPY [addr1][,addr2][,addr3][,num]
```

The arguments are:

addr1	Optional.
	Default: * (indicates the current statement)
	Specifies the address of the first statement to be copied.
addr2	Optional.
	Default: #1
	Specifies the address of the last statement to be copied or, if stated as a number, (e.g., #25), the number of statements to be copied.
addr3	Optional.
	Default: * (indicates the current statement)
	Specifies the address of the statement after which the copied statement(s) is to be placed. If "--" is specified, the copy is placed before (not after) the first line.
num	Optional.
	Default: 1
	Specifies the number of copies to be generated. Use this argument to repeat one or more statements.

Note that the COPY command uses the currently allocated SD work file to capacity before generating an error message. For example, if you request 10 copies of 8 lines and there is only space for 60 lines, the 60 lines will be created, after which an error message is generated: "SD FILE TOO SMALL".

## DEFINE Command

The DEFINE command enables you to change the default edit characteristics assumed by UED (as opposed to the QUERY command, which is used to display existing default values).

The command format is:

```
DEFINE key1=[value1][,key2=[value2]]...[,keyn=[valuen]]
```

where each *keyn* is one of the keywords described in the following text. Note that if you give a keyword argument with no value specified, the keyword is initialized to the standard default value.

Available keywords and their default values are:

CMDDEL	Optional.
	Default: CMDDEL=;
	Specifies the command delimiter character that is to be used to separate commands when entering multiple commands.

CON	Optional.
	Default: CON=YES
	Specifies whether or not the terminal device buffer size is the limitation factor that is to be placed upon the amount of output to be generated by any one command function.
	Possible values are YES and NO.
	CON=YES specifies that the buffer size for the terminal is to be considered as one output display. For a TTY device, the size is 2000 characters; for a 3277 model 2 device, 23 statements. To continue output, press ENTER.
	CON=NO specifies that all output will be written continuously.
DSN	Optional.
	Default: DSN=null value
	Specifies the name of the file to be used when the READ command function is executed.
	If this keyword is initialized to a file name, all READ commands entered without arguments obtain their input from this file.
	If this keyword is initialized to a null value, note that all READ commands entered must specify the source file that is to be used for input.
LISTCOL	Optional.
	Default: LISTCOL=73-80,1-72
	Specifies the order of the columns to be displayed when using the LIST command.
LRECL	Optional.
	Default: LRECL=80
	Specifies the logical record length that is to be assumed for the SD work file while editing. If a SAVE command is entered, the LRECL is not changed in the output file.
	Note that the value specified must be between 2 and 247. If an odd value is specified, the actual LRECL used is rounded to the next largest even value.
	<p><b>Notes:</b></p> <ol style="list-style-type: none"> <li>1. Padding with blanks occurs on both input and output files if the input and output files have an LRECL smaller than the LRECL specified for UED.</li> <li>2. Truncation occurs on both input and output files if LRECL is less than the LRECL of the input and output file.</li> </ol>

MEMB	Optional.
	Default: MEMB=null value
	Specifies the name of the member in a source library to be used for input to UED.
	If this keyword is initialized to a specific value, all READ command functions will default to this member name if the member name is not given in the READ command.
	If this keyword is initialized to null values and a source library is to be edited, note that all READ commands must specify a member name.
NUMCHAR	Optional.
	Default: NUMCHAR=#
	Specifies the character to be used as the prefix when command arguments as numbers are indicated.
OPDEL	Optional.
	Default: OPDEL=,
	Specifies the character to be used as the argument delimiter for commands.
SEQCOL	Optional.
	Default: SEQCOL=73-80
	Specifies the column numbers to be used for sequence numbers.
	Note that the value specified must be in the format nn-nn.
	SEQCOL=N indicates that sequence columns are not to be used.
SIZE	Optional.
	Default: SIZE=1500
	Specifies the number of records to be used when allocating the SD work file. The actual number of records used depends upon both the device type (3330, 3350, etc.) on which the SD data set is to be stored (full track utilization always occurs) and the LRECL value specified for the SD work file.
STRDEL	Optional.
	Default: STRDEL='
	Specifies the character to be used when character strings containing other delimiters (e.g., blanks, semicolons, commas, or a null string) are delimited.
TABCHAR	Optional.
	Default: TABCHAR=%
	Specifies the character to be used when setting tab stops for the terminal.

TABCOL	Optional.
	Default: TABCOL=10,16,41,72,73
	Specifies the default columns at which tabs are to be placed when the TAB command function is used.
	Note that the maximum number of tab settings that can be used is 10. The tabs must be specified one at a time.
	TABCOL=N indicates that tabs are not to be used.
TAGCHAR	Optional.
	Default: TAGCHAR=/
	Specifies the character to be used as the prefix when the data being edited for a given tag is searched.
TAGCOL	Optional.
	Default: TAGCOL=1-8
	Specifies the columns to be assumed by UED as the beginning and ending columns when a tag command function request is processed.
	Note that the value specified should be in the format n-n.
TRI	Optional.
	Default: No translation occurs
	Specifies the translation to occur on characters input from the terminal during editing.
	The value entered should be in the format:
	where the quotation marks must be entered. Each xx represents the hexadecimal format of a character input from the terminal. Each yy represents the hexadecimal format of the resulting translation.
TRO	Optional.
	Default: No translation occurs
	Specifies the translation that is to occur on characters output to the terminal during editing.
	The value entered should be in the format:
	where the quotes must be entered. Each xx represents the hexadecimal format of a character prior to output to the terminal. Each yy represents the hexadecimal format of the resulting translation on output.

VER	Optional.
	Default: VER=ON
	Specifies whether or not statement verification is to be on.
	Two values are permitted: ON and OFF.
	VER=ON indicates that every statement modified during editing is displayed at the terminal for modification verification. The current line is also displayed after every command that alters the current line pointer.
	VER=OFF indicates that statements modified during editing will not be displayed at the terminal for modification verification. The current line is not displayed when the current line pointer is changed.
VOL	Optional, except as noted below for VSE.
	Default: VOL=null value
	Specifies the volume identification number of the direct access volume on which a file being read with the READ command function will be located.
	Note:/tab/If this keyword is initialized to a specific volume, that volume and only that volume will be searched for the input READ command file.
	In MVS, if this keyword is initialized to a null value, the system catalog will be used to resolve which volume to locate the READ command file on (unless the volume is indicated in the READ command).
	In VSE, this keyword is required for the READ command.

## DELETE Command

The DELETE command enables you to delete one or more of the data statements that you are editing.

The command format is:

**DELETE [addr1][,addr2]**

The arguments are:

addr1	Optional.
	Default: * (indicates the current statement)
	Specifies the address of the first statement to be deleted.
addr2	Optional.
	Default: #1
	Specifies the address of the last statement in a range of statements to be deleted or, if stated as a number (e.g., #25), specifies the number of statements to be deleted.

If the current line is not deleted, the current line pointer is not changed; otherwise, the current line pointer is set to the address of the line immediately following the second argument address. One exception is when the second argument specifies the address of the last line; in this case, the current line pointer is set to the new last line.

To clear the SD work file without terminating UED, specify:

**DELETE --,++**

## END Command

The END command enables you to terminate the UED utility program.

### Note:

Termination of UED depends upon the status of the SD work file.

The command format is:

**END [n]**

where n is an optional argument with one of the following values:

I	Terminates UED immediately and deletes the SD work file.
H	Terminates UED immediately but holds or does not delete the SD work file. The SD work file may subsequently be recovered using the RECOVER command function of UED.

If you use the END command function with no arguments, UED is terminated only if the SD work file has not been modified since the last SAVE or SUBMIT function.

Note that use of either the CLEAR key on a 3270-type terminal, "control/D" on a TTY-type terminal, or the Com-plete program termination command \*EOJ is identical in function to use of the END command with no arguments.

## FIND Command

The FIND command enables you to find or locate one or more occurrences of a designated character string.

The command format is:

**FIND st[,num][,addr1][,addr2]**

The arguments are:

st	Required.
	Specifies the character string to be found.
num	Optional.
	Default: 1
	Specifies the number of occurrences of the designated character string to be found.
	ALL specifies that all occurrences are to be found.
addr1	Optional.
	Default: *+1
	Specifies the address of the statement in which the search is to begin.
addr2	Optional.
	Default: ++
	Specifies the address of the statement in which the search is to terminate or, if stated as a number (e.g., #25), specifies the number of statements that are to be searched.

The FIND command sets the current line pointer to the line where the last occurrence of the designated string was found. If the string is not found, no change in the line pointer will occur.

## HALT Command

The HALT command enables you to terminate command execution within UED or terminate output of UED (for example, at a hard copy terminal).

The command format is:

**HALT** [**xxx**]

where the argument *xxx* can be:

TYPING	Specifies that all UED output is suppressed until the end of the current command line is reached or until a RESUME command is entered.
	This argument can be abbreviated with T.
EXECUTION	Specifies that execution of the current UED command function (e.g., LIST) will be terminated and all other command functions entered as part of the same command line will be ignored.
	This argument can be abbreviated with X.

If you do not enter any arguments with the HALT command, the TYPING argument is taken as default.

If you specify CON=YES with the DEFINE command function, the HALT command is the only command function that terminates command execution at the end of one screen of terminal output when more output is to be displayed.

## HELP Command

The HELP command enables you to display syntax information about a given command or display a list of all commands available for use with UED.

The command format is:

```
HELP [command]
```

where the optional argument *command* represents any command name recognized by UED. If you enter a command name, a syntax description of that command is displayed. If you do not specify argument, a list of available command function names is generated.

You can optionally execute the HELP command by using the format:

```
? [command]
```

where the functions performed and the description of the command are the same as described above. The question mark (?) is a synonym of the character string HELP.

## INSERT Command

The INSERT command enables you to insert one or more lines after a designated line.

When you enter the INSERT command, you are placed in INSERT mode and remain in this mode until you enter a null statement (that is, press ENTER without entering additional data).

The command format is:

```
INSERT [addr][,S]
```

The arguments are:

addr	Optional.
	Default: * (indicates the current statement)
	Specifies the address of the statement after which the insertion will be made.
	Note that if "--" is specified, the insertion is made before (not after) the first line.
S	Optional.
	Default: None
	Specifies that, as each additional statement is added, the characters entered preceding the first blank statement are to become the sequence number. The characters after the first blank become the data starting in column one or, if the sequence columns start in column one, the first column after the sequence columns.

## LIST Command

The LIST command enables you to display one or more statements without changing the current line pointer.

The command format is:

```
LIST [addr1][,addr2]
```

The arguments are:

addr1	Optional.
	Default: * (indicates the current statement)
	Specifies the address of the first statement to be listed.
addr2	Optional.
	Default: #1
	Specifies the address of the last statement in the set of statements to be listed or, if stated as a number (e.g., #25), specifies the number of statements to be listed.

## MACRO Command

The MACRO command enables you to define a string of commands that are to be executed each time you enter the name of the macro definition.

### Note:

Macro statements are valid only for the duration of one execution of UED. As soon as UED terminates, all macro definitions for that execution will be lost.

*Each macro definition must fully reside in one statement. If you wish to use more than one statement, you must create multiple macro definitions.*

Macro statements, by definition, specify UED commands to be executed when the macro statement is issued by name. Since a macro statement is also a UED command, it can issue another macro statement, or even issue itself. Thus, you can chain macro statements together by having each macro execute one or more other macro statements.

Three command formats are available for a macro definition.

### DEFINE

The first MACRO command format enables you to define a specific macro statement.

The command format is:

```
MACRO DEFINE name cmd1[;cmd2]...[;cmdn];MEND
```

The arguments are:

DEFINE	Required.
	Specifies that the DEFINE function of the MACRO command is to be executed.
name	Required.
	Specifies the name by which this macro statement is to be known and referenced.
	Note that name must follow standard naming conventions (that is, eight characters or less, the first character being alphabetic).
cmd1	Required.
	Specifies the UED command to be executed when the macro name is used as a command function.
cmdn	Optional.
	Default: None
	Specifies one or more additional UED command functions to be executed when the macro name is entered as a command function.
	Note that each command function must be separated by the command delimiter defined by the CMDDEL keyword of the DEFINE command.
MEND	Required.
	Is a constant designating the end of the macro definition.

### Example:

The following example defines a macro command that, when executed, will list statements 200 through 400. In addition, 800 statements will be scanned beginning with statement 500, and a listing of those statements will be given for those containing the character string "programmer":

```
MACRO DEFINE A L 200,400;F PROGRAMMER,1,500,#800;MEND
```

You could then execute this macro command simply by entering:

```
A
```

### LIST

The second format for the MACRO command enables you to obtain a listing of all currently defined macro names or a listing of one or more macro definitions.

The command format is:

```
MACRO LIST [name1][,name2]...[,namen]
```

The arguments are:

LIST	Required.
	Specifies that the list option of the MACRO command is to be executed.
name	Optional.
	Default: All current macro names as defined by MACRO commands are listed.
	Specifies the name of a previously defined macro definition. If one or more macro names are given, each macro definition is listed.

### Example:

To obtain a list of all currently defined macro statements, you would enter:

```
MACRO LIST
```

The output of this request lists the macros currently defined by name.

To obtain a display of a specific macro definition (macro A in this example), issue the command:

```
MACRO LIST A
```

### DELETE

The third format for the MACRO command function enables you to delete a previously entered macro definition.

The command format is:

```
MACRO DELETE name[,namen]...
```

The arguments are:

DELETE	Required.
	Specifies that the delete option of the MACRO command function is to be executed.
name	Required.
	Specifies the name of the macro definition to be deleted.
namen	Optional.
	Specifies one or more additional macro definitions to be deleted.

### MOVE Command

The MOVE command enables you to move one or more statements within the data being edited.

The command format is:

```
MOVE [addr1][,addr2][,addr3]
```

The arguments are:

addr1	Optional.
	Default: * (indicates the current statement)
	Specifies the address of the first statement to be included in the move operation.
addr2	Optional.
	Default: #1
	Specifies the address of the last statement to be included in the move operation or, if stated as a number (e.g., #25), specifies the number of statements to be moved.
addr3	Optional.
	Default: * (indicates the current statement)
	Specifies the address of the statement after which the move or relocation will occur.
	Note that if "--" is specified, the move is made before (not after) the first statement.

If the current line address is not included in the range of statements you are moving, the current line address is not changed; otherwise, the current line address becomes that of the statement following the last statement indicated by the second argument (addr2).

## NUMBER Command

The NUMBER command enables you to change the statement sequence numbers for the statements being edited.

The command format is:

```
NUMBER [num1][,num2][,label][,N][,addr1][,addr2]
```

The arguments are:

num1	Optional.
	Default: 10
	Specifies the sequence number to be used for the first statement.
num2	Optional.
	Default: Same as that specified for num1.
	Specifies the increment to be used when the statements are numbered.
label	Optional.
	Default: None
	Specifies the label or identifier that is to be placed in the leftmost position of the sequence number.
	Note that label can be any identifier. If the constant DATE is chosen as a replacement for label, the current date is placed in the leftmost five positions of each sequence number in the format YYDDD.
N	Optional.
	Default: All statements will be renumbered.
	Specifies that only those statements that have been modified or inserted since the beginning of the session will be renumbered.
addr1	Optional.
	Default: --
	Specifies the address of the first statement that is to be renumbered.
addr2	Optional.
	Default: ++
	Specifies the address of the last statement in a range of statements to be renumbered or, if stated as a number (e.g., #25), indicates how many statements are to be renumbered.

## PRINT Command

The PRINT command enables you to obtain a hard copy printout of the data being edited.

The command format is:

```
PRINT [code][,format][,addr1][,addr2]
```

The arguments are:

code	Optional.										
	Default: The screen-to-hardcopy terminal assigned to the terminal in use (if any); otherwise, the default is the same terminal. The screen-to-hardcopy terminal can be defined in the Complete Terminal Information Block Table (TIBTAB) or altered by using the HC function of UUTIL.										
	Specifies the Terminal Identification number (TID) of the terminal to receive the printout or terminal name.										
format	Optional.										
	Default: FORMAT										
	Specifies the format to be used when printing.										
	This argument can have one of three values: FORMAT, CC, and NOCC.										
	FORMAT specifies that all output is arranged in pages, 60 statements per page, with a standard page header identifying the user ID, originating TID, library name, member name, date, and time.										
	Note that all pages are sequence numbered.										
	CC specifies that column one of each statement being edited contains a standard ASA carriage control character to control formatting of the printout. The standard ASA carriage control characters are:										
	<table border="0"> <tr> <td>blank</td> <td>advance 1 line and print</td> </tr> <tr> <td>0</td> <td>advance 2 lines and print</td> </tr> <tr> <td>-</td> <td>advance 3 lines and print</td> </tr> <tr> <td>1</td> <td>advance to a new page and print</td> </tr> <tr> <td>+</td> <td>suppress spacing</td> </tr> </table>	blank	advance 1 line and print	0	advance 2 lines and print	-	advance 3 lines and print	1	advance to a new page and print	+	suppress spacing
	blank	advance 1 line and print									
	0	advance 2 lines and print									
-	advance 3 lines and print										
1	advance to a new page and print										
+	suppress spacing										
Note that no header is provided and pages are not sequence numbered.											
NOCC specifies there are no carriage control characters in column one of the statements being edited. Page overflow printing is in effect and all statements print with no carriage control.											
addr1	Optional.										
	Default: --										
	Specifies the address of the first statement with which printing begins.										
addr2	Optional.										
	Default: ++										
	Specifies the address of the last statement in a range of statements to be printed or, if stated as a number (e.g., #25), specifies the number of statements to be printed.										

## QUERY Command

The QUERY command enables you to display the default UED editing characteristics as defined or defaulted by the DEFINE command.

The command format is:

```
QUERY [key][,key]...
```

The arguments are:

key	Optional.
	Default: If no key is given, a list of all default keyword values will be given.
	Represents a keyword argument, less the equal sign, as used in the DEFINE command. For each keyword argument listed, a display of the current value assigned to that argument will be given.
	As with the DEFINE command, the keywords can be abbreviated to the shortest number of characters that uniquely identifies them.

## READ Command

The READ command enables you to read or obtain data statements from a library for editing purposes.

The data being read can be a member of a source library or (in MVS) a sequential file. Note that if you specify a source library, you must either specify a member or default to the member initialized with the MEMB keyword in the DEFINE command.

You can read an entire member or portions thereof. All data read is placed in the SD work file, if it already exists, in the location that you specify. If the SD work file does not currently exist, a new one is created.

### Note:

If you issue a READ command, subsequent READ commands you issue without terminating UED will concatenate additional data statements to those originally read. To avoid this, as well as the need to terminate UED, make sure to delete all statements from the SD work file before you perform a subsequent READ by using the DELETE command.

*Before using the READ command, you must be familiar with the DSN, MEMB, and VOL keyword options of the DEFINE command function.*

The format of the READ function is:

```
READ [library][ (member) ] [ /volume ] [ , num1 ] [ , num2 ] [ , addr ]
```

Note that the first three argument values (library, member, and volume), if entered, must not be separated by spaces or commas.

The arguments are:

library	Optional.
	Default: None, unless the DEFINE command function is used to initialize the keyword DSN.
	Specifies the name of the input source library.
	If two-character library identification codes are used (as is done with the UEDIT utility program), the data set name need not be entered; simply enter the two-character code. Note, however, that these codes must be either defined to UED in the table of library identification codes (UEDTB1) or defined by the UL function of the UUTIL utility.
member	Optional.
	Default: None, unless the DEFINE command function is used to initialize the keyword MEMB.
	Specifies the member in a source library to be used as input.
	Note that only the first parenthesis is required.
volume	Optional.
	Default: None, unless the DEFINE command function is used to initialize the keyword VOL.
	Specifies the volume identification number of the direct access volume to be searched for the library.
	Note that the slash (/) must be included.
	In MVS, if this argument is not used and no default value is defined, the system catalog is used to locate the file.
num1	Optional.
	Default: 1
	Specifies the relative statement number (relative to one) of the first statement to be included in the read operation.
num2	Optional.
	Default: ALL
	Specifies the number of statements to be read.
addr	Optional.
	Default: ++
	Specifies the address of the statement within the current SD work file after which the statement(s) being read will be placed. Note that if "--" is entered, the statements read are placed in front of the first statement line.

## Example

The following example illustrates the use of the READ command. In this example, the Com-plete initialization procedure "COMPLETE" is read from the library SYS1.PROCLIB and only the first three statements of the procedure are to be read:

```
READ SYS1.PROCLIB(COMPLETE),1,3
```

## RECOVER Command

The RECOVER command enables you to resume editing of the data in a specific SD work file after either a system failure occurs or after UED terminates with the H option of the END command function.

If Com-plete is initialized with the accounting option in effect, or if you log onto Com-plete before the initial editing session, the SD work file is recovered based upon your user ID.

If Com-plete is initialized without the accounting option in effect, and if you do not log on to Com-plete before the initial editing session, the SD work file is recovered based upon the Terminal Identification (TID) number. In this situation, you can only recover from the original terminal.

### Note:

The RECOVER command only works if you use it before you edit any other file or member.

The command format is:

```
RECOVER
```

## REPEAT Command

The REPEAT command enables you to repeat the execution of the previously entered command line.

### Note:

You cannot issue a REPEAT command if the previous input command line contains either a REPEAT command or a macro statement containing a REPEAT command.

The command format is:

```
REPEAT
```

## RESUME Command

The RESUME command enables you to resume output or typing after it has been halted by a HALT command.

The command format is:

```
RESUME
```

## SAVE Command

The SAVE command enables you to save or write the data being edited to an output source library member or library. All or some of the statements being edited may be saved.

The command format is:

```
SAVE [library][(member)][/volume][,addr1][,addr2]
```

The arguments are:

library	Optional, except as indicated below.
	Default: The library originally used in the initial READ command function, if any. If this is new data being entered (that is, a READ command function was not entered), an output library must be specified.
	Specifies the name of the output library to be used when the SAVE function is performed.
	If two-character library identification codes are used (as is done with the UEDIT utility program), the data set name need not be entered; simply enter the two-character code. Note, however, that these codes must be defined to UED in the table of library identification codes (UEDTB1) or defined by the UL function of the UUTIL utility.
member	Optional, except as indicated below.
	Default: The member name specified in the READ command function initially executed, if any. If this is a new member being entered (that is, a READ command function was not entered), and an output source library is used, an output member must be specified.
	Specifies the name of the output member to be used when the save operation is performed.
	Note that only the first parenthesis is required.
volume	Optional, except as indicated below.
	Default: None. If the output library is not cataloged, this argument must be specified.
	Specifies the volume identification number of the direct access volume to be searched for the library.
	Note that the slash (/) must be included if this argument is used.
	In MVS, if this argument is not used, the system catalog is used to locate the file. If no arguments are given, however, the file included in the original read is not cataloged, and the save occurs in the same library.
	Note that if a library code is used when the library is referenced in a read and the library code table contains the volume serial number, the volume argument is not required.
addr1	Optional.
	Default: --
	Specifies the address of the first statement to be included in the save operation.

addr2	Optional.
	Default: ++
	Specifies the address of the last statement to be included in the save operation or, if stated as a number (e.g., #25), specifies how many statements are to be saved.

**Note:**

If you use the SAVE command function to save only a range of statements rather than the entire SD work file, the range of statements entirely replaces an existing member (that is, the existing member is not updated).

**SUBMIT Command**

The SUBMIT command enables you to submit a range of statements via the RJE facility of Com-plete to be scheduled for batch execution.

You can submit either the entire SD work file or portions thereof for scheduling.

The command format is:

```
SUBMIT [addr1][,addr2]
```

The arguments are:

addr1	Optional.
	Default: --
	Specifies the address of the first statement in the SD work file to be submitted.
addr2	Optional.
	Default:++
	Specifies the address of the last statement within the SD work file to be included in the submit operation or, if stated as a number (e.g., #25), specifies how many statements to submit.

**TAB Command**

The TAB command enables you to display an edit scale that indicates the available columns in which you can perform editing. This scale also contains the character T in every tab position.

The command format is:

```
TAB
```

No arguments are required or provided.

The output displayed by this command is illustrated in the following figure:

```
T. |...8 |...|...1T0...T..2|0...|...3|0... (...)
```

The following notes apply to the items in the preceding display:

- Columns 72-80 are listed first. These are the sequence number columns that normally appear in the output of a LIST command.
- A space separates the sequence columns from the data columns.
- The character T is entered in every column for which a tab is set; in this example, columns 72, 10, and 16.
- Column numbers divisible by five are indicated with a vertical bar (|). Column numbers divisible by ten are indicated by having the numerical digits representing those columns separated by a vertical bar (e.g., column 30 is represented as "3|0", where the 3 is actually in column 29 and the 0 is actually in column 31).

## Forward Paging Command

The Forward Paging command enables you to advance the current line pointer address forward a designated number of lines.

The command format is:

**+num**

where *num* is any integer larger than zero.

This command causes the current line pointer address to move forward the indicated number of lines.

## Backward Paging Command

The Backward Paging command enables you to decrement the current line address a designated number of lines.

The command format is:

**-num**

where *num* is any integer larger than zero.

This command causes the current line pointer address to move backward the indicated number of lines.

## Last Statement Paging Command

The Last Statement Paging command enables you to advance the current line pointer address to the last statement.

The command format is:

**++ [-num]**

Use of the *-num* argument positions to "num" lines before the last statement (e.g., ++-5 would position to five lines before the end of file).

## First Statement Paging Command

The First Statement Paging command enables you to change the current line pointer address to the first statement.

The command format is:

```
--[+num]
```

Use of the *+num* argument positions you to "num" lines after the first statement (e.g., `++5` would position to five lines after the top of file).

## Tag Paging Command

The Tag Paging command enables you to change the current line pointer address to that of the statement containing a designated tag. A tag is a character string ranging from 1 to *n* characters.

The search for a tag is normally in columns one through eight of each statement; however, you may override this with the TAGCOL keyword option of the DEFINE command function.

The command format is:

```
/tag[+num]
```

where you must enter the slash (/) before the tag itself.

Use of the *+num* or *-num* argument positions you to "num" lines after or before, respectively, the tag statement (e.g., `/tag+5`, `/tag-5`).

## Statement Modification Command

The Statement Modification command enables you to advance the current line pointer address forward to a designated statement and optionally modify the contents of that statement.

The command format is:

```
num[data]
```

where *num* is any string representing the desired statement sequence number.

Note that data (that is, any characters specified after the sequence number) will replace the data currently contained in the statement identified by the specified statement number.