Line Commands

This section describes the line commands provided by the program editor. A line command always applies to the source line in which you enter it or to a block of lines marked by multiple line commands.

Depending on the configuration of your installation, line commands can be entered in lower case. In this section, however, all commands are shown in upper case to distinguish them as commands.

▶ **To execute a line command**

1. In the leftmost column of the prefix area, next to the required source line, type a line command over the characters contained in these columns, and press ENTER.

   It is not possible to enter a line command if either of the following applies:

   - Insert mode is active (indicated by a series of apostrophes (' ' ' ' ' ')). In this case, toggle the insert mode to non-insert mode.

   - The prefix area is protected. In this case, deactivate protection with the PROTECT editor command.

   Or:
   In the command line, enter a line command preceded by a colon (:), place the cursor in the line to which you want to apply the command, and press ENTER. For example:

   In the command line, enter :I, place the cursor in the first line and press ENTER. A new line is then inserted below the first line.

2. If required, enter the RESET editor command to reset a pending line command and delete all line labels.

Command	Function
)	Moves this line right by two columns.
)n	Moves this line right by n columns, irrespective of any other text in the line: you may lose text in the moved line.
))n	Marks the first line of a block of lines to be moved right by n columns. A second))n is required to mark the last line of the block. The function is performed when the second))n is specified. The block is moved regardless of any other text in the block: you may lose text in the moved block.
(	Moves this line left by two columns.
(n	Moves this line left by n columns regardless of any other text: you may lose text in the moved lines.
((n	Marks the first line of a block of lines to be moved left by n columns. A second ((n is required to mark the last line of the block. The function is performed when the second ((n is specified.
<	Moves the text in this line left by two columns.
>	Moves the text in this line right by two columns.

Command	Function
`>n`	Moves the text in this line right by n columns or up to last non-blank character: no text is lost.
`>>n`	Marks the first line in a block of lines to be moved to the right by n columns or up to last non-blank character. A second `>>` is required to mark the last line of the block. The function is performed when the second `>>` is specified.
`<n`	Moves the text in this line left by n columns or up to first non-blank character.
`<<n`	Marks the first line in a block of lines to be moved to the left by n columns or up to first non-blank character. A second `<<` is required to mark the last line of the block. The function is performed when the second `<<` is specified.
`A`	Marks the target line for a move (`M`, `Mn`, `MM`) or copy (`C`, `Cn`, `CC`) line command. The moved or copied line(s) are inserted *after* this line.
`B`	Marks the target line for a move (`M`, `Mn`, `MM`) or copy (`C`, `Cn`, `CC`) line command. The moved or copied line(s) are inserted *before* this line.
`BNDS`	Displays the boundary positions in this line. See also the `BNDS` editor command.
`C`	Copies this line to the position indicated by an `A`, `B` or `O` line command.
`Cn`	Copies this line and the next `n-1` lines to the position indicated by an `A`, `B` or `O` line command.
`CC`	Marks the first line of a block of lines to be copied. A second `CC` command is required to mark the last line of the block to be copied. The function is performed when the second `CC` is specified. The lines are copied to the position indicated by an `A`, `B` or `O` line command.
`CX`	Copies the line labeled with `.X`. Inserts text after this line.
`CY`	Copies the line labeled with `.Y`. Inserts text after this line.
`CX-Y`	Copies the block of lines from the line labeled with `.X` to the line labeled with `.Y`. Inserts text after this line.
`COLS`	Displays the column positions in this line.
`D`	Deletes this line.
`Dn`	Deletes the current line and the next `n-1` lines.
`DD`	Marks the first line of a block to be deleted. A second `DD` command is required to mark the last line of the block to be deleted. The deletion is performed when the second `DD` is entered.
`DX`	Deletes the line labeled with `.X`.
`DY`	Deletes the line labeled with `.Y`.
`DX-Y`	Deletes the block of lines from the line labeled with `.X` to the line labeled with `.Y`.
`F`	Includes the first excluded line.
`Fn`	Includes the first n excluded lines.

Command	Function
I	Inserts one line. The editor switches to insert mode indicated by ´´´´´´. This means if you type text or enter a blank in the new line and press ENTER, a new line is automatically inserted and the cursor placed in it. If you enter no new text in an inserted line and press ENTER, the editor leaves insert mode and the blank line is deleted. You can also fill an inserted line with a predefined content (see the MASK editor command).
I*n*	Inserts *n* lines. You can type text in the new lines. When you press ENTER, unused lines are deleted but one blank line remains with the cursor in it (editor stays in insert mode).
.I(*obj,pos,n*)	Inserts any object contained in the current library in the current system file into the editor screen. **Note:** This command is entered in the first position of a source line, not in the prefix area. The *pos* entry indicates the line at which the include operation is to begin. For example, setting *pos* to 20 causes the insertion to begin with the 20th line in the source. The *n* entry indicates the number of lines to be inserted. If the object is a Natural map, an INPUT USING MAP statement (see *INPUT Syntax 2 - Using Predefined Map Layout* in the *Statements* documentation) with all defined variables is automatically included in the current line. If the object is a data area, the entire data area is included, except comment lines. Only data areas that have been saved and cataloged with the STOW system command, can be included in the source contained in the editing area. If the object is an adapter, a PROCESS PAGE USING (see *Syntax 2 - PROCESS PAGE USING* in the *Statements* documentation) with all defined variables is automatically included in the current line.
.I(*)	Invokes a selection list of objects in the current library.
J	Joins the next line with this one. You can specify how many of the characters of the following line are to be joined by placing the cursor at the point in the line where it is to be separated and pressing ENTER. To join the entire line, place the cursor outside the line to be joined. This command is identical to the TJ command.
L*n*	Includes the last *n* excluded lines.
LC	Changes the letters in this line to lower case.
LC*n*	Changes the letters in this line and in the next *n-1* lines to lower case.

Command	Function
LCC	Marks the first line of a block of lines in which to change all letters to lower case. A second LCC is required to mark the last line in the block. The function is performed when the second LCC is specified.
LJ	Justifies the text within the set boundaries in this line with the left boundary.
LJJ	Marks the first line of a block of lines within the set boundaries to be justified to the left. A second LJJ command is required to mark the last line of the block to be justified. The justification is performed after the second LJJ command has been issued.
M	Moves this line to the position indicated by an A, B or O line command.
M*n*	Moves this line and the next *n-1* lines to the position indicated by an A, B or O line command.
MM	Marks the first line of a block of lines to be moved. A second MM command is required to mark the last line of the block to be moved. The function is performed when the second MM is specified. The lines are moved to the position indicated by an A, B or O line command.
MASK	Inserts a blank line in the editing area in which you can define a mask. This line is inserted whenever the I*n* line command is used to create one or more new lines. See also the MASK editor command and *To define and use a mask line*.
MX	Moves the line labeled with .X. Inserts it after this line.
MY	Moves the line labeled with .Y. Inserts it after this line.
MX-Y	Moves the block of lines from the line labeled with .X to the line labeled with .Y. Inserts it after this line.
N	Modifications made in this line do not take effect when ENTER is pressed.
NZ	Places this line at the top of the editing area when the POINT editor command is issued.
O	Marks this line as the target line for a move (M, M*n*, MM) or copy (C, C*n* CC) line command. The moved or copied line(s) are merged with this line, that is, blank characters in the line are overlaid.
O*n*	Marks this line and the next *n-1* lines as the target lines for a move (M, M*n*, MM) or copy (C, C*n* CC) line command. The moved or copied lines are merged with these lines, that is, blank characters in the lines are overlaid.
OO	Marks the first line of a block of target lines for a move (M, M*n*, MM) or copy (C, C*n* CC) line command. A second OO command is required to mark the last line of the block of target lines. The moved or copied line(s) are merged with these lines, that is, blank characters in the lines are overlaid.
R	Repeats this line once.
R*n*	Repeats this line *n* times.

Command	Function
RR	Marks the first line of a block of lines to be repeated. A second RR command is required to mark the last line of the block to be repeated. The repeat operation is performed when the second RR is entered.
RR*n*	Repeats the block of lines *n* times.
RJ	Justifies the text within the set boundaries in this line with the right boundary.
RJJ	Marks the first line of a block of lines within the set boundaries to be justified to the right. A second RJJ command is required to mark the last line of the block to be justified. The justification is performed when the second RJJ is entered.
S	Splits this line into two lines beginning at the cursor position. Type in the command, move the cursor to the position where the line is to be split, and press ENTER.
T	Scrolls the source to make the marked line the top line.
TABS	Displays the tab positions in this line. See also the TABS editor command.
TC	Centers the text within the set boundaries in this line.
TCC	Marks the first line of a block of lines within the set boundaries to be centered. A second TCC command is required to mark the last line of the block of the centered. The centering is performed when the second TCC command is entered.
TE	Excludes all lines below this line from display and switches to insert mode. The excluded lines appear again when you press ENTER.
TF	Joins this line with the following lines up to the next blank line. The bounds settings can be used to restrict the columns affected (see the BNDS editor command).
TF*n*	This line command can be entered with a numerical value specifying the right boundary. For example, the line command TF5 orders text with column 5.
TI	Inverts the sequence of all characters in the current line and within the set boundaries.
TII	Marks the first line of a block of lines to be inverted within set boundaries. Requires a second TII to mark the last line of the block. The function is performed when the second TII is specified.
TJ	Joins the next line with this one. Same as the J line command.
TO	Joins this line with the next one.
TOO	Marks the first line of a block of lines within the set boundaries to be joined. A second TOO command is required to mark the last line of the block to be joined. The function is performed when the second TOO is entered.
TS	Splits this line into two lines at the cursor position; a blank line is also automatically inserted, but deleted if unused (identical to the S line command).
UC	Changes all letters in this line to upper case.
UC*n*	Changes all letters in this line and in the next *n-1* lines to upper case.

Command	Function
UCC	Marks the first line of a block of lines in which to change all letters to upper case. A second UCC is required to mark the last line of the block. The function is performed when the second UCC is specified.
W	Opens a window with one line.
W*n*	Opens a window with *n* lines.
WC	Copies the data window. The cursor position marks the column at which this line is to be split to insert the copied text.
WC*n*	Splits this line in column *n*, and copies the text between the two parts of the line.
WE	Marks the end of the data window. Works in the same way as WS. If the window is to start and end in the same line, replace the WS command by the WE command. The editor acknowledges the set window with message WW in the prefix area. See also *Copying and Moving Text with a Data Window*.
WM	Moves the data window. Works in the same way as WC, but the original text is deleted after the copy operation. See also *Copying and Moving Text with a Data Window*.
WM*n*	Splits this line in column *n*, and moves the text between the two parts of the line.
WS	Marks the start of the data window. The cursor position marks the column from which text is read. If the cursor is not in the line for which the command is entered, column 1 is taken. See also *Copying and Moving Text with a Data Window*.
WS*n*	Specifies that the data window starts in column *n* of this line.
X	Excludes this line from display.
X*n*	Excludes this line and the following *n* lines from display.
XX	Marks the first line of the block of lines to be excluded from display. A second XX is required to mark the second line of the block. The function is performed when the second XX is specified.
.X	Marks this line with .X. See also *.label* below.
.Y	Marks this line with .Y. See also *.label* below.
.label	Marks this line with *.label* where *label* is a string of 1 to 4 alphabetic characters. For example: line command .X names this line .X, and .Y names this line .Y. See also the LABEL editor command.