Program Editor

The Natural program editor is used to perform online full-screen editing of Natural source programs. With the program editor, you can create and edit source programs quickly and efficiently with a minimum of effort. This section describes how to use this editor:

Invoking the Program Editor

You invoke the program editor with the system command EDIT as described in the Natural Command Reference documentation.

When you invoke the program editor, the editor screen is displayed (as shown below with a program in the work area):

```
>                                      > +  Program     SAGDEMO  Lib SAGTEST
 All   ....+....1....+....2....+....3....+....4....+....5....+.Mode Structured.
   0010 DEFINE DATA LOCAL USING L-INVOIC
   0020             LOCAL USING L-INV-LN
   0030 END-DEFINE
   0040 *
   0050 READ INVOICE-VIEW BY INVOICE-NO FROM 1
   0060 *
   0070   FIND INVOICE-LINE-VIEW WITH INVOICE-NO = INVOICE-NO (0050)
   0080     DELETE
   0090   END-FIND
   0100 *
   0110   DELETE
   0120   END TRANSACTION
   0130 END-READ
   0140 *
   0150 FETCH 'MENU'
   0160 END
   0170
   0180
   0190
   0200
         ....+....Current Source Size: 308 Char. Free: 44756   +.. S 16   L 1
```

Note:
If Natural ISPF is installed and the general editor profile default "ISPF Editor as Program Editor" is set to "Y", instead of the program editor, either the Natural ISPF main menu (if the EDIT command is entered without an object name) or the Natural ISPF editor screen with the specified object is invoked.

Top Information Line

The top information line of the editor screen is used to display a message indicating object modification. See also the section Source Status Message. In addition, the programming mode (structured or reporting) currently in effect is displayed. When a program is read into the edit area, the mode is set to the one which was in effect when the program was stowed. This information is only displayed if the "Source Size Information" parameter in the editor profile defaults is set to "Y".

Bottom Information Line

In the bottom information line of the editor screen, the following items of information are displayed:

Current Source Size	Size (number of characters) of the current object. As source lines are stored in variable length in the work area, trailing blanks within a source line are not counted; leading and embedded blanks are counted. This information is only displayed if the "Source Size Information" parameter in the editor profile defaults is set to "Y".
Char. Free	The number of characters still available in the work area. This information is only displayed if the "Source Size Information" parameter in the profile defaults is set to "Y".
S	Size (number of lines) of the object being edited.
L	The number of the source line currently displayed as the top line.

Editor Command Line

The top line of the program editor screen is the edit command line. In this line, you can enter:

- a Natural system command (for example, EDIT, CHECK, SAVE),
- one or more editor commands,
- the name of a Natural program to be executed.

Additionally, the following items of information are displayed:

Direction Indicator (+ or -)	The direction indicator can be set to control the direction of the editor commands ADD and SCAN and of the line commands ".C", ".I" and ".M". The value "+" indicates **after** and the value "-" indicates **before**. The exact interpretation is described with the relevant command description.
Object Type	The type of object currently in the work area. The object type can be changed by using the editor command SET TYPE.
Object Name	The name of the object currently in the work area.
Library (Lib)	The library to which you are currently logged on.

Editing a Program

Multiple Functions

Multiple functions can be performed from a single input screen:

- Source lines can be updated directly.
- One or more line commands can be used.
- One or more editor commands can be used.

The following restrictions related to multiple functions apply:

- Only one insert line command (.I) can be performed at a time.
- You can enter multiple commands in the command line of the editor: you can enter more than one editor command, but only the last command entered in the editor command line can be a Natural system command. For example:
  SC ’MOVE’,-2,RENUMBER.

- If you have changed the screen contents manually or by commands, a system command cannot be entered until you press ENTER.

Note:
Natural treats the editor command "N" like a system command.

Dynamic Conversion from Lower to Upper Case

When the Natural terminal command %L is set and dynamic conversion to upper case is specified, all source code you enter in the editor is automatically converted to upper case, with the following exceptions:

- Text strings that are not hexadecimal constants and are enclosed in apostrophes remain as you enter them.
- Text strings (with or without apostrophes) in objects of type Text remain as you enter them.
- Comments remain as you enter them.

Dynamic conversion from lower to upper case can be specified and deactivated in the editor profile.

Editor Commands

Editor commands are entered in the command line of the program editor. The command parameters must be separated either by the input delimiter character as defined with the Natural session parameter ID (the default delimiter character is comma ",") or by a blank. When multiple commands are entered, these must also be separated by the delimiter character or by blanks. Line commands must not be entered in the command line.

The following edit commands are available:

Editor Command	Function
ADD[(*n*)]	This command adds *n* blank lines. If the direction indicator is set to "+", the lines are added after the last line of the object being edited; if the direction indicator is set to "-", the lines are added before the first line of the object. The value for *n* can be in the range from 1 to 9. If *n* is not (or not correctly) specified, 9 lines (4 in split-screen mode) are added by default. With the next ENTER, lines that are still left blank will be eliminated.
CANCEL	With this command you leave the editor. Any modifications made since the last time the SAVE command was entered are **not** saved.
CHANGE	This command scans for the value entered as *scan-value* and replaces each such value found with the value entered as *replace-value*. The syntax for this command is: CHANGE *'scan-value 'replace-value '* Any special character which is not valid within a Natural variable name can be used as the delimiter character.
CLEAR	This command clears the edit area (including the line markers "X" and "Y").
DX, DY, DX-Y	This command deletes the X-marked line; or the Y-marked line; or the block of lines delimited by "X" and "Y". See also the line commands ".X" and ".Y".
EX, EY, EX-Y	This command deletes source lines from the top of the source area to, but not including, the X-marked line; or from the source line following the Y-marked line to the bottom of the source area; or all source lines in the source area excluding the block delimited by "X" and "Y". See also the line commands ".X" and ".Y".
EXIT	With this command you leave the editor.
LET	This command undoes all modifications made to the current screen since the last time ENTER was pressed. In addition, LET ignores all line commands already entered but not yet executed.
N [(*n*)]	This command renumbers the source code lines of the program currently in the work area. If you only enter "N", the lines are numbered in increments of 10; if you enter "N (*n*)", the lines are renumbered in increments of *n*. If the value specified for *"n"* is too big, lines are numbered in increments of 5. **Note:** See also Renumbering of Source-Code Line Number References.
PROFILE [*name*]	This command displays the current editor profile.
QUIT	Same as editor command CANCEL.

Editor Command	Function
REN ON|OFF	ON Renumbers a Natural source program whenever it is checked, run, saved, stowed or cataloged. OFF Indicates that automatic renumbering is not in effect. The default is ON (see also the section Editor Profile). **Note:** See also Renumbering of Source-Code Line Number References.
RESET	This command deletes the current X and Y line markers and any marker previously set with the line command ".N". See also line commands ".X" and ".Y".
SCAN ['*scan-value*']	This command scans for data in the source area. If you enter SCAN without any parameter, the SCAN menu is invoked. If you enter SCAN '*scan-value*', a scan for *scan-value* is performed. If the supplied scan value is entered without delimiter characters, for example, "SCAN ABC D", the entire character string which follows the keyword SCAN is used as the scan value. SCAN is a cursor-sensitive command.
SCAN = [+|-]	This command scans for the next occurrence of the scan value. The direction of the scan operation is determined by the setting of the direction indicator. If the direction indicator is omitted or set to "+", the scan operation will be from the current position of the source area (top of the displayed source window) to the last line in the source area. If the direction indicator is set to "-", the scan operation will be backwards from the bottom line of the current screen to the first line in the source area. The direction for a given scan command can also be explicitly specified by entering "SCAN =+" or "SCAN =-" prior to command execution. The first line which contains the scanned value is positioned to the top line after each SCAN command. Each line in which the scanned value is located is marked with an "S" to the left of the line. **Note:** The equal sign "=" used with the SCAN command is the default input assign character. If another character has been specified as input assign character (see session parameter IA as described in the Natural Parameter Reference documentation), that other character must be used instead.
SET ABS [ON|OFF]	ON The SCAN and CHANGE commands operate in absolute mode, which means that the value to be scanned/changed need not be delimited by blanks or special characters. OFF The SCAN and CHANGE commands do not operate in absolute mode, which means that the value to be scanned/changed must be delimited by blanks or special characters. The default is OFF.
SET ESCAPE *character*	The escape character which must precede each line command. The default escape character is ".".

Editor Command	Function
SET NUL [ON|OFF]	ON All occurrences of a value scanned with the SCAN command are deleted. After the deletion of the scanned value, the SET NUL command is automatically set to OFF. The default is OFF.
SET RANGE [ON|OFF]	ON The SCAN and CHANGE commands operate in range mode, which means that the value to be scanned/changed must be located within the range of lines delimited by the X and Y line markers. OFF The SCAN and CHANGE commands operate in non-range mode, which means that no range limit is to be in effect. The default is OFF.
SET SEQ [ON|OFF]	OFF If your input is numeric, the first four positions in the edit area are considered as the line number and are moved to the line number position once you press ENTER. This feature is useful, for example, if a statement line is to be referenced by a source code line number in another statement line; when you renumber the source code, the referencing line number is renumbered, too. ON Numeric input in the first four positions remains as entered. Except with object type Text, the default is OFF.
SET SIZE [ON|OFF]	ON The program size is displayed at the bottom information line of the editor screen and the programming mode is displayed on the scale line. OFF This information is not displayed. The default is OFF.
SET STAY [ON|OFF]	ON The current screen will stay when ENTER is pressed. Forward and backward positioning can be done by positioning commands only. OFF Pressing ENTER positions to the next screen.
SET TYPE	The object type is set automatically when an existing object is read into the work area. This command can be used to change the type of object to be edited: SET TYPE PROGRAM (Natural Program) SET TYPE SUBROUTINE (Natural Subroutine) SET TYPE SUBPROGRAM (Natural Subprogram) SET TYPE HELPROUTINE (Natural Help Routine) SET TYPE COPYCODE (Natural Copycode) SET TYPE TEXT (Natural Text) SET TYPE CLASS (Natural Class)
SHIFT [-|+*nn*]	This command shifts each source line delimited by the X and Y markers to the left or right. The *nn* parameter represents the number of characters the source line is to be shifted. Comment lines are not shifted.
SHIFT - -	This command shifts each source line delimited by the X and Y markers to the leftmost position. Comment lines are not shifted.

Editor Command	Function
SHIFT ++	This command shifts each source line delimited by the X and Y markers to the rightmost position (maximum 99 positions). Comment lines are not shifted.
STRUCT [DISPLAY]	This command performs structural indentation of a Natural source program. If DISPLAY is specified, the Natural source program is displayed in compressed form (see also the system command STRUCT in the Natural Command Reference documentation).
*	This command displays the editor command most recently entered.
*=	This command again executes the command most recently entered in the command line.
.	With this command you leave the editor. Any modifications made since the last time the SAVE command was entered are **not** saved.

See also Renumbering of Source-Code Line Number References in the Natural Reference Documentation.

Editor Commands for Positioning

Editor commands for positioning are entered in the command line of the program editor. The following commands are available for positioning:

Command	Function
+P	Position forwards one page.
+	
-P	Position backwards one page.
-	
+H	Position forwards half a page.
-H	Position backwards half a page.
T	Position to top of program.
- -	
B	Position to bottom of program.
++	
+*nnnn*	Position forwards *nnnn* lines (maximum 4 digits).
-*nnnn*	Position backwards *nnnn* lines (maximum 4 digits).
nnnn	Position to line number *nnnn*.
X	Position to the line marked with "X".
Y	Position to the line marked with "Y".
POINT	Positions to the line in which the line command ".N" was entered. See also the line command ".P".

Line Commands

The line commands are listed below. The notation "(*nnnn*)" indicates a repetition factor. The default repetition value is 1 (with the exception of the ".I" command; see below).

Note:
You are recommended to enter a blank at the end of each line command. This prevents the editor from attempting to interpret any information existing on the line as part of the line command.

Line Command	**Function**
.C(*nnnn*)	Copies the line in which the command was entered.
.CX(*nnnn*) .CY(*nnnn*)	Copies the X-marked or the Y-marked line. See also the line commands ".X" and ".Y" as well as the notes in the following section.
.CX-Y(*nnnn*)	Copies the block of lines delimited by the X and Y markers. (See also the notes in the following section.)
.D(*nnnn*)	Deletes line or lines. The default is 1 line.
.I(*n*)	Inserts *n* empty lines, where *n* can be in the range from 1 to 9. If *n* is not (or not correctly) specified, 9 lines (4 lines in split-screen mode) are inserted by default. (See also the notes in the following section.)
.I(*obj*,*ssss*,*nnnn*)	Includes into the source an object contained in the current library or in the steplib (the default steplib is SYSTEM). Depending on the direction indicator, the object is inserted before or after the line in which you enter the command. If you wish to include only part of the object, you specify as *ssss* the first line to be included (e.g., "20" means the inclusion will start from the 20th line), and as *nnnn* the number of lines to be included. If you enter multiple commands, this command is always executed after all other line and/or editor commands have been executed. If the object is a map, an INPUT USING MAP statement 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 stowed local and parameter data areas can be included into the source area; global data areas cannot be included.
.J	Joins the current line with the next line. If the resulting line exceeds the length of the editor screen line, the line is marked with "L" and must be split in two with the ".S" command (see below) before it can be modified.
.L	Undoes all modifications that have been made to the line since the last time ENTER was pressed.
.MX .MY	Moves the X-marked or the Y-marked line. See also the line commands ".X" and ".Y" as well as the notes below.
.MX-Y	Moves the block of lines delimited by the X and Y markers (see also the notes below).

Line Command	Function
.N	Marks (invisibly) a line to be positioned to the beginning of the source area by the editor command POINT. The mark is automatically deleted when an error with a line command or editor command occurs.
.P	Positions the line marked by this command to the top of the screen.
.S	Splits the line at the position marked by the cursor.
.X	Marks a line or the beginning of a block of lines, to be processed (see also the notes below).
.Y	Marks a line or the end of a block of lines, to be processed (see also the notes below).

Note:
If both the commands ".X" and ".Y" are applied to one line, it is treated as being marked with "X" and with "Y"; the line marker actually shown to reflect this status is a "Z".

If the direction indicator is set to "+", the copied, inserted or moved lines are placed after the line in which the corresponding command was entered; if the direction indicator is set to "-", the copied, inserted or moved lines are placed before the line in which the command was entered.

Special PF-Key Functions

The following special functions can also be controlled using PF keys:

Function	Explanation
*CURSOR	A line split function can be combined with the command ".I", ".CX", ".CX-Y", ".MX" or ".MX-Y". This is accomplished by assigning the value "*CURSOR" to a PF key. If this PF key is then pressed instead of ENTER after a line command has been entered, the line in which the command was entered is first split at the cursor position and then the line command is executed.
*X *Y	If a PF key is assigned the value "*X" or "*Y", the cursor position is marked X or Y whenever this PF key is used. These column markers are then used to determine which portion of a line is to be included in the command operation. See the example below.

Example:

```
                  XY
X       0010 MOVE A TO B
        0020 WRITE A B
Y       0030 MOVE B TO A
                  XY
        ....
        0100 .MX-Y.....
```

The block of text starting with the "A" in line 0010 and ending with the "B" in line 0030 is moved:

```
0010 MOVE
0030 TO A
....
0010 A TO B
0020 WRITE A B
0030 MOVE B
```


Cursor-Sensitive Commands

- The SCAN Commands
- The SPLIT Command
- The EDIT and LIST System Commands

Cursor-sensitive commands are commands where, instead of entering a name in the command line, you can mark the name with the cursor anywhere on the editor screen (except in the command line). You can place the cursor on any word that is not in the command line. It does not matter where on the word the cursor is placed.

The SCAN Commands

The SCAN [*'scan-value'*] command scans for data in the edit area. If the SCAN command is used without any parameter but with the cursor positioned outside the editor command line, this results in a scan operation for the string the cursor is positioned to. If the cursor is positioned to a blank character, however, the SCAN menu is invoked.

In split-screen mode, the cursor can be positioned to a string in the split-screen area, too. The scan operation, however, is performed in the edit area only.

When using the SPLIT SCAN [*'scan-value'*] command, the same applies as for the SCAN command, but the scan operation is performed in the split-screen area only (see also the section Split-Screen Commands).

Note:
To benefit from cursor sensitiveness as much as possible, the SCAN or SPLIT SCAN command should be assigned to a PF key.

The SPLIT Command

Instead of the commands SPLIT PROGRAM, SPLIT DATA, SPLIT FUNCTION and SPLIT VIEW, which you can use to display a programming object or DDM in the split-screen area of the editor (see also the section Split-Screen Commands), you only have to enter the command SPLIT and place the cursor on the name of the desired object. The object must be contained in the current library.

Note:
To benefit from cursor sensitiveness as much as possible, the SPLIT command should be assigned to a PF key.

The EDIT and LIST System Commands

The system commands EDIT and LIST are cursor-sensitive, too. Instead of specifying an object name, the cursor can be positioned to a text string of the object currently in the edit area that corresponds to the desired object name.

With the EDIT command, the corresponding object is loaded into the editor. If necessary, even a different editor is invoked.

With the LIST command, the corresponding object is listed, even if a view has been referenced.

For more information on EDIT and LIST see the Natural Command Reference documentation.

The Exit Function

If the editor default parameter "Prompt Window for Exit Function" is set to "Y", any time you enter the EXIT command in the command line, the EXIT Function prompt window is invoked, offering you the following options:

Option	Explanation
Save and Exit	Leaves the editor and saves all modifications made to the current object.
Exit without Saving	Leaves the editor without saving any modification made to the current object since the last SAVE command was entered.
Resume Function	Neither leaves the editor nor saves any modifications; the prompt window is closed and the current function is resumed.

When "Prompt Window for Exit Function" is set to "N", the EXIT command leaves the editor and saves all modifications made to the current object; no prompt window is displayed.

