

INPUT

- Syntax 1 - Dynamic Screen Layout Specification
- Syntax 2 - Using Predefined Map Layout

Related Statement: REINPUT

See also Screen Design / Windows in the Natural Programming Guide.

Function

The INPUT statement is used in interactive mode to create a formatted screen or map for data entry. It may also be used in conjunction with the Natural stack (see the STACK statement); and on mainframe computers, it may also be used to provide user data for programs being executed in batch mode.

Input Modes

The INPUT statement may be used in screen, forms, or keyword/delimiter mode. Screen mode is generally used with video terminals/screens. Forms mode may be used with TTY terminals. Delimiter mode is used with TTY terminals, and also in batch mode (on mainframe computers). The default mode is screen mode.

You can change the input mode with the session parameter IM or the terminal commands %F and %D.

Screen Mode

In screen mode, execution of the INPUT statement results in the display of a screen according to the fields and positioning notation specified. The message line of the screen is used by Natural for error messages. The position of the message line (top or bottom of screen) may be controlled by the terminal command %M. The terminal user may position to specific fields using the various tabulation keys.

As Natural allows for screen window processing, the layout of the logical screen map may be larger (theoretically 250 characters per line and 250 lines, but limited by the internal screen buffer) than the physical screen size.

The windowing terminal command %W may be used to modify logical and physical window position and size (see the terminal command %W for details of window handling).

For input fields (AD=A or AD=M) that are not fully displayed on the physical screen, the following rules apply:

- Input fields whose beginning is not inside the window are always made protected.
- Input fields which begin inside and end outside the window are only made protected if the values they contain cannot be displayed completely in the window. Please note that in this case it is decisive whether the value length, not the field length, exceeds the window size. Filler characters (as specified with the profile parameter FC or session parameter AD) do not count as part of the value.
- Before an input field thus protected can be accessed and processed, the window size must be adjusted so as to fully display the field or value respectively (see the terminal command %W).

Non-Screen Modes

The INPUT statement may be used for an operation on line-oriented devices or for the processing of batch input from sequential files.

The same map layouts as defined for screen mode operation can also be processed in non-screen mode.

Forms mode and keyword/delimiter mode are also available to process the input either by simulating the screen layout in line mode or by just processing the data without any map layout.

See also:

Using the INPUT Statement in Non-Screen Modes | Using the INPUT Statement in Batch Mode on Mainframe Computers | Processing Data from the Natural Stack

Entering Data in Response to an INPUT Statement

Data for an alphanumeric field must be entered left-justified. Any character, including a blank, is meaningful. The data are assigned one character per byte to the internal field. Data entered for an alphanumeric field are not validated.

Lower and upper case translation are controlled by the terminal commands %L and %U as well as the attributes AD=T and AD=W.

Data for a numeric field may be placed anywhere in the input field. Leading and/or trailing blanks, leading zeros, a leading sign and one decimal point are permitted. Natural adjusts the value according to the internal definition of the field. If SG=OFF is specified, Natural does not assume or allocate a position for a sign position. Data for a field defined with format P must be entered in decimal form. Natural will convert decimal to packed wherever necessary. A field containing all blanks is interpreted as a zero value. Data for a numeric field are validated by Natural to ensure that the value consists only of leading and/or trailing blanks, an optional leading sign, an optional decimal point, and numeric characters. If no decimal point is entered, it is assumed to be to the right of the value entered.

Data for a binary field must be entered for all positions (two characters per byte). Only valid hexadecimal characters (0 - 9, A - F) may be used. A blank (H'20' in ASCII or H'40' in EBCDIC respectively) is valid and is converted to binary zeros. Data for a binary field are validated by Natural for hexadecimal characters.

Data for format L fields may be entered as blank ("false") or non-blank ("true").

Data for format F, D, and T are entered according to the rules stated for F, D, and T constants.

Numeric Edit Mask Free Mode

Within a field element, you may format the representation of the field content with an edit mask. The edit mask is used for two purposes:

- to build the layout for displaying the field on the screen;
- when a string has been modified and ENTER has been pressed, to extract the field data from the string entered.

The advantage of improving the format of the field data displayed with additional insert characters may actually be a disadvantage, because a new data value entered has to perfectly match the format of the edit mask.

Example:

```
SET GLOBALS ID=; DC=,  
RESET N (N7,3)  
INPUT N (AD=M EM=Z'.' 'ZZZ'.' 'ZZZ,999EUR)  
END
```

Output value	is displayed as:	Input value	must be entered as:	leads to Input error NAT1145 if entered as:
0	,000EUR	1	1,000EUR	1 1EUR 01,000EUR
1234	1.234,000EUR	1234567	1.234.567,000EUR	1234567 1.234.567 1.234.567EUR
0,123	,123EUR	1,234	1,234EUR	1,234

Another option for entering numeric fields with the edit mask is to use an alternative INPUT mode, which is called the edit mask free mode. When activated (either at session Startup with the profile parameter EMFM or in a running Natural session via the terminal command %FM+), all or some of the edit mask insert characters may be left out from input.

However, when a contiguous string of insertion characters appears in the edit mask (like "EUR" in the example below), you may only supply or leave out the string completely. The number of optional or mandatory digits (edit-mask character "Z" and "9") to be supplied is not affected.

Example with Edit Mask Free Mode activated:

```

SET GLOBALS ID=; DC=,
SET CONTROL 'FM+' /* activate numeric Edit Mask Free Mode
RESET N (N7,3)
INPUT N (AD=M EM=Z'.'ZZZ'.'ZZZ,999EUR)
END
    
```

Input value	can be entered as:	leads to error NAT1145 if entered as:
1	1 1,0 001 1,00EUR 0.001 1,EUR	1EUR
1234567	1234567 1.234.567 1234.567 1234567,0 1.234.567,0 1.234.567,EUR 1.234.567,0EUR 1.234.567,000EUR	1.234.567EUR 1.234.567,000EUR
1,234	1,234 1,234EUR 001,234 0.001,234EUR 00001,234EUR	1,234EU

Note:

The edit mask free mode applies only for INPUT, but is ignored in a MOVE EDITED statement.

SB - Selection Box

Selection boxes in an INPUT statement are available on mainframe computers only. For other platforms, selection boxes may be defined in the map editor only.

Selection boxes can be attached to input fields. They are a comfortable alternative to help routines attached to fields, since you can code a selection box direct in your program. You do not need an extra program as with help routines.

For more information, see SB in the Session Parameters documentation.

Error Correction

If the value entered in an input field does not correspond to the format or edit mask of the field, Natural displays an error message (without terminating the program execution) and positions the cursor in the field in error. The user may then enter a valid value, whereupon processing continues.

Split-Screen Feature

In general, each INPUT statement generates a new page (or terminal screen) of output. Any INPUT statement which is specified within an AT END OF PAGE statement will not produce a new screen. This feature allows for the creation of a split screen where the upper portion of the screen may be used to display multiple lines and the lower portion can be used to create an input map for communication. The PS parameter (page size) should be used, either in a SET GLOBALS or FORMAT statement, to set the logical page size to ensure that the input map is built on the same physical screen.

The first INPUT line will be placed after the last displayed line. If the NO ERASE option is used, the first INPUT line will be placed at the top of the page.

Syntax 1 - Dynamic Screen Layout Specification

INPUT [WINDOW = 'window-name'] [NO ERASE] [(statement-parameters)] [WITH-TEXT-option] [MARK-option] [ALARM-option]		
$\begin{bmatrix} nX \\ nT \\ x/y \end{bmatrix}$	$\begin{bmatrix} \text{'text' [(attributes)]} \\ \text{'c'(n) [(attributes)]} \\ \text{'-' } \\ \text{'=' } \\ \text{'/... } \end{bmatrix}$	$\begin{bmatrix} *IN \\ *OUT \\ *OUTIN \end{bmatrix} \{operand1 [(parameter)]\} \dots$

Operand	Possible Structure	Possible Formats	Referencing Permitted	Dynamic Definition
Operand1	S A G N	A N P I F B D T L G	yes	yes

This form of the INPUT statement is used to create a layout of an INPUT screen, or to create an INPUT data layout which is to be read (on mainframe computers) in batch mode from a sequential input file.

INPUT WINDOW='window-name'

With the option WINDOW='window-name', you indicate that the INPUT statement is to be executed for the specified window. The specified window must be defined in a DEFINE WINDOW statement. The specified window is only active for the duration of that INPUT statement, and is automatically deactivated when the INPUT statement has been executed.

See also the statements DEFINE WINDOW and SET WINDOW.

NO ERASE

NO ERASE causes a screen map of an INPUT statement to be overlaid onto an existing screen without erasing the screen contents.

Screen as used here refers to a logical screen rather than a physical screen.

All unprotected fields that existed on the screen are converted to protected (display only) fields. The old data remain on the screen until the new layout is displayed. If a field from the new screen content partially overlays an existing field, the one character before the new field and the next character in the existing field will be replaced by a blank.

statement-parameters

One or more parameters, enclosed within parentheses, may be specified immediately after the INPUT statement or an element being displayed.

Each parameter specified in this manner will override any previous parameter specified in a GLOBALS command, SET GLOBALS or FORMAT statement. If more than one parameter is specified, one or more blanks must be present between each entry. An entry may not be split between two statement lines.

The parameter settings applied here will only be regarded for variable fields, but they have no effect on text-constants. If you would like to set field attributes for a text-constant, they have to be set explicitly for this element.

Parameters that can be specified with the INPUT statement		Specification S = at statement level E = at element level
AD	Attribute Definition	SE
AL	Alphanumeric Length for Output	SE
BX	Box Definition	SE
CD	Color Definition	SE
CV	Control Variable	SE
DF	Date Format	SE
DY	Dynamic Attributes	SE
EM	Edit Mask	SE
FL	Floating Point Mantissa Length	SE
HE	Helproutine	SE
IP	Input Prompting Text	SE
LS	Line Size	S
MC	Multiple-Value Field Count	S
MS	Manual Skip	S
NL	Numeric Length for Output	SE
PC	Periodic Group Count	S
PM	Print Mode	SE
PS	Page Size	S
SB	Selection Box	E
SG	Sign Position	SE
ZP	Zero Printing	SE

The individual parameters are described in the section Session Parameters of the Natural Parameter Reference documentation.

Example:

```

DEFINE DATA LOCAL
1 VARI (A4)      INIT <'1234'>          /*   Displays
END-DEFINE      /*   as
FORMAT AD=M    /*   -----
INPUT          'Text'          VARI      /*   Text 1234
INPUT (PM=I)   'Text'          VARI      /*   Text 4321
INPUT          'Text' (PM=I)    VARI (PM=I) /*   txeT 4321
INPUT          'Text' (PM=I)    VARI      /*   txeT 1234
END

```

Examples of using parameters at the statement and element level are provided below.

1. Display attributes (see the session parameter AD in the Natural Parameter Reference documentation).
2. Color attributes (see the session parameter CD in the Natural Parameter Reference documentation).

Dynamic Replacement of Message Text - operand3

Operand3 represents a numeric or text constant or the name of a variable.

The values provided are used to replace parts of the message text.

The notation ":n:" is used within the message text as a reference to *operand3* contents, where "n" represents the *operand3* occurrence (1 - 7).

Example:

```

...
MOVE 'MESSAGE-1' TO #FIELD
...
INPUT WITH TEXT 'THE ERROR IS :1: ',#FIELD ...
...
    
```

This would cause the following message to be output:

THE ERROR IS MESSAGE-1

Note:

Multiple specifications of operand3 must be separated from each other by a comma. If the comma is used as a decimal character (as defined with the session parameter DC) and numeric constants are specified as operand3, put blanks before and after the comma so that it cannot be misinterpreted as a decimal character.

Alternatively, multiple specifications of operand3 can be separated by the input delimiter character (as defined with the session parameter ID); however, this is not possible in the case of ID=/ (slash), because the slash has a different meaning in the INPUT statement syntax.

Insignificant zeros or blanks will be removed from the field value before it is displayed in a message.

MARK-option

```

MARK [POSITION operand4 [IN]] [FIELD] { operand1 }
                                     { *fieldname }
    
```

Operand	Possible Structure			Possible Formats												Referencing Permitted	Dynamic Definition	
Operand4	C	S					N	P	I								yes	yes
Operand1	C	S	A				N	P	I								yes	yes

Field to be Marked - operand1

Each AD=A or AD=M (that is, non-protected field) specified in an INPUT statement is assigned a field reference number, beginning with 1. The MARK option positions the cursor to the field number specified. The **fieldname* notation may be used to position to a field using the name of the field as a reference.

MARK POSITION

With MARK POSITION, you can have the cursor placed at a specific position - as specified with *operand4* - within a field. *Operand4* must not contain decimal digits.

Examples:

```
MARK 3
MARK #A
MARK *#
MARK POSITION 3 IN #A
```

ALARM-option

```
[ [AND] [SOUND] ALARM]
```

This option causes the sound alarm feature of the terminal to be activated when the INPUT statement is executed. The appropriate hardware must be available to be able to use this feature.

Default Prompting Text

Unless the session parameter IP (input prompting) is set to IP=OFF, the field name of the field used in an INPUT statement will be displayed preceding the field value (forms mode) or as a prompting keyword to select the field (keyword/delimiter mode). This default field name may be overridden by specifying either a 'text' element (which replaces the default name) or '-' (which suppresses the display of the default field name) immediately preceding the field name.

Field Positioning, Text Specification, Attribute Assignment

```

[ nX ] [ 'text' [(attributes)] ] [ *IN ]
[ nT ] [ 'c' (n) [(attributes)] ] [ *OUT ] {operand1 [(parameter)]} ...
[ x/y ] [ ' - ' ] [ *OUTIN ]
[ / ... ]
```

Several notations are available for field positioning, attribute assignment, and text creation.

nX

Causes *n* spaces to be inserted between fields.

Note: (for Mainframes only)

This notation inserts *n* spaces between columns. *n* must not be "0".

nT

Causes positioning (tabulation) to print position *n*.

x/y

Places the next element on line *x*, beginning in column *y*. *y* must not be "0". Backward positioning in the same line is not permitted.

'text'

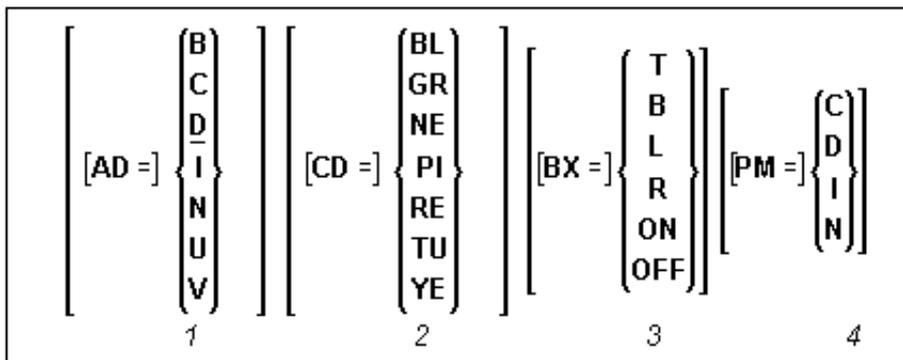
Causes text to be displayed write protected.

'c' (n)

Identical to *'text'*, except that the character *c* is displayed *n* times. *n* must be 1 - 132.

attributes

Indicates the attributes to be used for display. The following *attributes* may be used:



1. Display attributes (see the session parameter AD in the Natural Parameter Reference documentation).
2. Color attributes (see the session parameter CD in the Natural Parameter Reference documentation).
3. Outlining attributes (see the session parameter BX in the Natural Parameter Reference documentation).
4. Print mode attributes (see the session parameter PM in the Natural Parameter Reference documentation).

Minus Sign '-'

When placed before a field, '-' suppresses the generation of a field name as prompting text.

Note:

Any text string before a field will replace the field name as prompting text.

Equal Sign '='

When placed before a field, '=' results in the display of the field heading followed by the field contents.

Slash Sign '/'

When placed between fields or text elements, "/" causes positioning to the beginning of the next print line.

The contents of fields may be specified for input, output only, and output for modification using the attribute settings AD=A, AD=O, and AD=M respectively. The default is AD=A. All fields specified with AD=A (input only) or AD=M (output for modification) will create unprotected fields on the screen. A value for such a field may be entered

by the user.

For TTY devices, output for modification fields will occupy twice the size of the field (one for output, one for input) so that a new value may be entered. An input field (AD=A/M) specified as non-displayable will always start on a new line on a TTY device.

Example:

INPUT #A (AD=A) #B (AD=O) #C (AD=M)

#A is an input field which is unprotected, i.e., a value is to be entered for the field.

#B is a field which is to be displayed write-protected, i.e., no value may be entered for the field.

#C is a field whose current value is to be displayed, and the value may be modified by entering a new value for the field.

***IN, *OUT and *OUTIN**

Equivalent to the attributes AD=A, AD=O, AD=M respectively.

Field Specification - operand1

Operand1 represents the field to be used. Database fields or user-defined variables may be specified.

Natural directly maps the content of each field from the data area to the INPUT statement, no move operation is necessary.

When the content of a database field is modified as a result of INPUT processing, only the value as contained in the data area is modified. Appropriate database UPDATE/STORE statements must be used to change the content of the database.

When the name of a group of database fields is referenced in an INPUT statement, all fields belonging to that group will be individually used as input fields.

When reference is made to a range of occurrences within an array, all occurrences are individually processed as input fields, but no prompting text will be created for each individual occurrence, only for the first one.

parameters

One or more parameters, enclosed within parentheses, may be specified immediately after operand1 (see table and example below).

Each parameter specified will override any previous parameter specified in a GLOBALS command, SET GLOBALS or FORMAT statement. If more than one parameter is specified, they must be separated by one or more blanks from one another. Each parameter specification must not be split between two statement lines.

The parameter settings applied here will only be regarded for variable fields, but they have no effect on text-constants. If you would like to set field attributes for a text-constant, they have to be set explicitly for this element.

For information on the individual parameters, see the table in the section statement-parameters.

Note:

The session parameter EM will be referenced dynamically in the DDM if an edit mask is defined for a database field. Edit masks may be specified for output and input fields. When an edit mask is defined for an input field, the data for the field must be entered according to the edit mask specification.

Example 1 - Syntax 1

```

/* EXAMPLE 'IPTEX1': INPUT
/*****
DEFINE DATA LOCAL
1 #PNUM (A8)
1 #FNC (A1)
END-DEFINE
/*****
INPUT 10X 'SELECTION MENU FOR EMPLOYEES SYSTEM' /
      10X '-' (54) //
      10X 'ADD ' '(A)' /
      10X 'PURGE' '(P)' /
      10X 'UPDATE' '(U)' /
      10X 'TERMINATE' '(A)' ///
      10X 'PLEASE ENTER FUNCTION: ' #FNC
/*****
DECIDE ON EVERY VALUE OF #FNC
  VALUE 'A'
    WRITE 'Add function selected'
    /* invoke the object containing the Add function here
  VALUE 'P'
    WRITE 'Purge function selected'
    /* invoke the object containing the Purge function here
  VALUE 'U'
    WRITE 'Update function selected'
    /* invoke the object containing the Update function here
  VALUE '.'
    STOP
  NONE
    REINPUT 'PLEASE ENTER A VALID FUNCTION' MARK *#FNC
END-DECIDE
/*****
END

```

```

SELECTION MENU FOR EMPLOYEES SYSTEM
-----

```

```

ADD (A)
PURGE (P)
UPDATE (U)
TERMINATE (.)

```

```

PLEASE ENTER FUNCTION:

```

Example 2 - Syntax 1

```

/* EXAMPLE 'INPEX1': INPUT WINDOW
*
DEFINE WINDOW WIND1
  SIZE 10 * 40
  BASE 5 / 10
  FRAMED ON POSITION TEXT
*
INPUT WINDOW='WIND1' 'PLEASE ENTER HERE:'
  / #STRING(A15)
*
END

```

```

> r                                     > + Program      INPEX1   Lib SYSEXRM
All   ....+....1....+....2....+....3....+....4....+....5....+....6....+....7..
0010 /* EXAMPLE 'INPEX1': INPUT WINDOW
0020 *
0030 D +-----Top+
0040  ! PLEASE ENTER HERE:          !
0050  ! #STRING                      !
0060  !                               !
0070 * !                             !
0080 I !                             !
0090  !                               !
0100 * !                             !
0110 E !                             !
0120  +-----Bottom+
0130
0140
0150
0160
0170
0180
0190
0200
      ....+....1....+....2....+....3....+....4....+....5....+... S 11   L 1

```

Example 3 - Syntax 1

```

/* EXAMPLE 'INPEX2': INPUT WINDOW
*
ASSIGN #START (A30) = 'EXAM_'
*
INPUT (AD=M) MARK POSITION 5 IN *#START
  / 'PLEASE COMPLETE START VALUE FOR SEARCH'
  / 5X #START
END

```

```

PLEASE COMPLETE START VALUE FOR SEARCH
#START EXAM[ ]

```

Syntax 2 - Using Predefined Map Layout

```

INPUT [WINDOW = 'window -name'] [WITH-TEXT-option]
         [MARK-option]
         [ALARM-option]
         [USING] MAP map-name [NO ERASE]
         [ operand1 ... ]
         [NO PARAMETER ]
    
```

Operand	Possible Structure			Possible Formats												Referencing Permitted	Dynamic Definition		
Map-name	C	S			A													yes	yes
Operand1	C	S	A		A	N	P	I	F	B	D	T	L	C				yes	yes

This form of the INPUT statement is used to perform input processing using a map layout that has been created using the Natural map editor.

Map layouts can be used in two ways:

- the program does not provide a parameter list;
- the program does provide a parameter list (*operand1*).

INPUT USING MAP Without Parameter List

In this case the *map-name* must be specified as an alphanumeric constant (up to 8 characters).

The map used in this manner must have been created prior to the compilation of the program which references the map.

The names of the fields to be processed are taken dynamically from the map source definition at compilation time. The field names used in both program and map must be identical.

All fields to be referenced in the INPUT statement must be accessible at that point.

In structured mode, fields must have been previously defined (database fields must be properly referenced to processing loops or views). In reporting mode, user-defined variables may be newly defined in the map.

When the map layout is changed, the programs using the map need not be recataloged. However, when array structures or names, formats/lengths of fields are changed, or fields are added/deleted in the map, the programs using the map must be recataloged.

The map source must be available at program compilation; otherwise the INPUT USING MAP statement cannot be compiled. If you wish to compile the program even if the map is not yet available, specify NO PARAMETER: the INPUT USING MAP can then be compiled even if the map is not yet available.

INPUT Fields Defined in the Program

By specifying the names of the fields to be processed within the program (*operand1*), it is possible to have the names of the fields in the program differ from the names of the fields in the map.

The sequence of fields in the program must match the map sequence. Please note that the map editor sorts the fields as specified in the map in alphabetical order by field name. For more information, see the map editor description in your Natural Editors documentation.

The program editor line command ".I(*mapname*)" can be used to obtain a complete INPUT USING MAP statement with a parameter list derived from the fields defined in the specified map.

When the layout of the map is changed, the program using the map need not be recataloged. However, when field names, field formats/lengths, or array structures in the map are changed or fields are added or deleted in the map, the program must be recataloged.

A check is made at execution time to ensure that the format and length of the fields as specified in the program match the fields as specified in the map. If both layouts do not agree, an error message is produced.

INPUT WINDOW='window-name'

This option is described under **Syntax 1** of the INPUT statement.

WITH TEXT/MARK/ALARM-options

These options are described under **Syntax 1** of the INPUT statement.

USING MAP

USING MAP invokes a map definition which has been previously stored in a Natural system file using the map editor.

The *map-name* may be a 1- to 8-character alphanumeric constant or user-defined variable. If a variable is used, it must have been previously defined. The map name may contain an ampersand (&); at execution time, this character will be replaced by the current value of the Natural system variable *LANGUAGE. This feature allows the use of multi-lingual maps.

The execution of the INPUT statement causes the corresponding map to replace the current contents of the screen, unless the NO ERASE option is specified, in which case the map will overlay the current contents of the screen.

NO ERASE

This option is described under **Syntax 1** of the INPUT statement.

Field Specification - operand1

A list of database fields and/or user-defined variables, all of which must have been previously defined. The fields must agree in number, sequence, format and length with the fields in the referenced map; otherwise, an error occurs.

When the content of a database field is modified as a result of INPUT processing, only the value as contained in the data area is modified. Appropriate database UPDATE/STORE statements must be used to change the content of the database.

Using the INPUT Statement in Non-Screen Modes

Forms Mode

The terminal command %F causes forms mode to be in effect.

In forms mode, Natural will display all output text of the map layout on the terminal field by field according to the positioning parameters. This permits the user to enter data on a field by field basis. When all data are entered, the hardcopy output is produced exactly as it would have appeared on the screen.

In forms mode, entering %R permits the operator to retype the entire form in case of an error. The input is processed as in the first execution of the INPUT statement.

Keyword/Delimiter Mode

The terminal command %D causes keyword/delimiter mode to be in effect.

Data can be entered using keywords or positional input values.

Using keyword input, the terminal operator may enter data for the individual fields using the prompting text that, in forms mode, would have been displayed before the value as a keyword to identify the field. The keyword must be followed by the input assign character (IA parameter), followed immediately by the data. Any spaces following the assign character are taken as data up to the delimiter character (ID parameter). A delimiter character is not required after the last data element. Keyword data for the different fields may be entered in any order separated by the delimiter character. If the operator types in a keyword which is not defined in the INPUT statement, an error message will be returned. Data need not be entered for all input fields. Fields for which no data are entered are set to blank for alphanumeric fields and zero for numeric and hexadecimal fields.

Using positional value input, the terminal operator enters only data for all input fields separated by the currently defined input delimiter character (ID parameter). The sequence of fields for input must correspond to the sequence of the fields in the INPUT statement.

The user may switch from positional to keyword input by entering a number of values in positional input separated by the delimiter character and then switching to keyword mode for selected fields by specifying keywords in front of the values.

After a keyword has been used to position to a field, any non-keyword input following the keyword will be processed as positional input to be assigned to fields following the previously selected field in the INPUT statement.

Note:

A keyword and the corresponding input field must be on the same logical line. If their aggregate length exceeds the line size, adjust the line size (LS parameter) accordingly so that keyword and field fit onto one line.

Data entered in keyword/delimiter mode are validated as for screen mode. An error message will be returned if an attempt is made to enter more characters than defined for a field.

If the INPUT statement is to be processed in keyword/delimiter mode on a buffered (3270-type) terminal or a workstation, all data to be assigned to one INPUT statement must be entered on one screen. ENTER is only to be used when all data to the INPUT statement have been entered.

Processing Data from the Natural Stack

Data elements that have been placed in the Natural stack via a FETCH, RUN or STACK statement will be processed by the next INPUT statement encountered for execution.

The INPUT statement will process the data in keyword/delimiter mode as described above.

If data elements are not available to fill all input fields, fields will be filled with blank/zero depending on the field format. If more data elements are specified than input fields exist, the remaining data are ignored.

When a field is filled with data from the stack, the field attributes do not apply to the data.

The Natural system variable *DATA may be referenced to determine the number of data elements currently available in the Natural stack.

Using the INPUT Statement in Batch Mode on Mainframe Computers

Forms Mode

In batch forms mode, the INPUT map is displayed. A data record is read for each line containing one or more AD=A and/or AD=M fields, and the data contained in the record are assigned to the appropriate field (or fields).

Input data fields are assumed to be contiguous. Unless the delimiter character is used, input data must be entered in the exact length according to the internal definition of the field. For numeric fields, space must be allowed for a sign (if SG=ON) and decimal point when appropriate.

Data may optionally be entered using the delimiter character to separate the values of the individual fields. In this case, data need not be entered in the exact number of positions according to the internal definition but are processed from left to right beginning in position 1. The rules for data entry are the same as described under Entering Data in Response to an INPUT Statement. In addition, the assign character may be used to specify that the contents of an *OUTIN field are not to be reset.

Keyword/Delimiter Mode

Keyword/delimiter mode, when used in batch mode, functions the same as keyword/delimiter mode in TP mode with the following exceptions:

- The entire input map may be printed under the control of the terminal command %Q.
- *OUTIN fields retain their original values unless explicitly changed.

Use of Terminal Commands in Batch Mode

The following Natural terminal commands may be used when using the INPUT statement in batch mode on a mainframe computer:

Command	Explanation
%*	Record Suppression. When entered in position one and two of a record, %* causes the printing of the next input record to be suppressed. DATA RECORD %* SUPPRESSED DATA RECORD
%	Record Continuation. When % is entered as the last non-blank character of a record, the next input record will be treated as a continuation record. DATA, RECORD, WITH, CONTINUATION, % CONTINUATION RECORD INPUT V1 V2 V3 V4 V5 V6 DISPLAY V1 V2 V3 V4 V5 V6 will produce the following output: DATA RECORD WITH CONTINUATION CONTINUATION RECORD
%/	End-of-file. When entered in the first two positions of a record (without any trailing non-blank characters), %/ causes an end-of-file condition.
%%	Set restart point in input data stream.
%. .	Reading of input values for the current INPUT statement will be terminated.
% K <i>nn</i>	Simulate PF keys.
% KP <i>n</i>	Simulate PA keys.
% Q	This command causes printing of maps used to read input data to be suppressed.

See the Terminal Commands documentation for further information.

Additional JCL statements are required when using the INPUT statement for data entry in batch mode. The Natural administrator should be contacted to ensure that these statements have been provided before attempting to execute Natural in batch mode.