## EXAMINE

This chapter covers the following topics:

- Syntax 1 - EXAMINE
- Syntax 2 - EXAMINE TRANSLATE
- Syntax 3 - EXAMINE for Unicode Graphemes
- Examples

Related Statements: ADD | COMPRESS | COMPUTE | DIVIDE | MOVE | MOVE ALL | MULTIPLY | RESET | SEPARATE | SUBTRACT

Belongs to Function Group: Arithmetic and Data Movement Operations

## Syntax 1 - EXAMINE

```
EXAMINE [FULL [VALUE [OF]]] {\begin{array}{l}{\mathrm{ operandl }}\\{\mathrm{ SUBSTRING (operand1,operand2,operand3)}}\end{array}};}
[FOR][FULL [VALUE [OF]]] [PATTERN] operand4
[DELIMITERS-option]
{[DELETE-REPLACE-clause] [GIVING-clause]}
```

For an explanation of the symbols used in the syntax diagram, see Syntax Symbols.

## Syntax Description - Syntax 1

The EXAMINE statement is used to observe the content of an alphanumeric or binary field, or a range of fields within an array, and to

- return the number of how many times a search-pattern was found;
- return the byte position where a search-pattern appears first;
- return the significant content length of a field; that is, the field length without trailing blanks;
- return the occurrence number (indices) of an array field, where a pattern was found first;
- replace a pattern by another pattern;
- delete a pattern.

Operand Definition Table:

| Operand | Possible <br> Structure |  |  |  | Possible Formats |  |  |  |  |  |  | Referencing Permitted | Dynamic <br> Definition |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| operandl | C* | S | A | A | U |  |  |  | B |  |  | yes | no |
| operand2 | C | S |  |  |  | N | P | I | B* |  |  | yes | no |
| operand3 | C | S |  |  |  | N | P | I | B* |  |  | yes | no |
| operand4 | C | S |  | A | U |  |  |  | B |  |  | yes | no |

* operandl can only be a constant if the GIVING clause is used, but not if the DELETE/REPLACE clause is used.
* Format B of operand2 and operand3 may be used only with a length of less than or equal to 4 .

Syntax Element Description:
$\left.\begin{array}{|l|l|}\hline \text { operand1 } & \begin{array}{l}\text { operandl is the field whose content is to be examined. } \\ \text { If operandl is a DYNAMIC variable, a REPLACE operation may cause } \\ \text { its length to be increased or decreased; a DELETE operation may cause } \\ \text { its length to be set to zero. The current length of a DYNAMIC variable } \\ \text { can be ascertained by using the system variable *LENGTH. }\end{array} \\ \hline \text { operand4 } & \begin{array}{l}\text { operand4 is the value to be used for the examine operation. }\end{array} \\ \hline \text { FULL } & \begin{array}{l}\text { If FULL is specified for an operand, the entire value, including trailing } \\ \text { blanks, will be processed. If FULL is not specified, trailing blanks in the } \\ \text { operand will be ignored. }\end{array} \\ \hline & \begin{array}{l}\text { Normally, the content of a field is examined from the beginning of the } \\ \text { field to the end or to the last non-blank character. } \\ \text { With the SUBSTRING option, you examine only a certain part of the } \\ \text { field. After the field name (operandl) in the SUBSTRING clause, you } \\ \text { specify first the starting position (operand2) and then the length } \\ \text { (operand3) of the field portion to be examined. }\end{array} \\ \begin{array}{l}\text { For example, to examine the 5th to 12th position inclusive of a field \#A, } \\ \text { you would specify: }\end{array} \\ \begin{array}{l}\text { EXAmINE SUBSTRING (\#A, 5, 8) . } \\ \text { Notes: }\end{array} \\ \text { 1. If you omit operand2, the starting position is assumed to be 1. } \\ \text { 2. If you omit operand3, the length is assumed to be from the starting } \\ \text { position to the end of the field. } \\ \text { 3. If SUBSTRING is used in conjunction with a DYNAMIC variable, } \\ \text { the field behaves like a fixed length variable; that is, the length } \\ \text { (*LENGTH) does not change as a result of the EXAMINE } \\ \text { operation, regardless of whether a DELETE or REPLACE } \\ \text { operation was executed or not. }\end{array}\right\}$
$\left.\left.\begin{array}{|l|l|}\hline \text { PATTERN } & \begin{array}{l}\text { If you wish to examine the field for a value which contains "wild } \\ \text { characters", that is symbols for positions not to be examined, you use } \\ \text { the PATTERN option. operand4 may then include the following } \\ \text { symbols for positions to be ignored: }\end{array} \\ \text { - A period (.), question mark (?) or underscore (_) indicates a single } \\ \text { position that is not to be examined. } \\ \text { - An asterisk (*) or a percent sign (\%) indicates any number of } \\ \text { positions not to be examined. }\end{array}\right\} \begin{array}{l}\text { Example: With PATTERN 'NAT*AL' you could examine the field for } \\ \text { any value which contains NAT and AL no matter which and how many } \\ \text { other characters are between NAT and AL (this would include the values } \\ \text { NATURAL and NATIONAL as well as NATAL). }\end{array}\right\}$

## DELIMITERS Option



Operand Definition Table:

| Operand | Possible <br> Structure |  |  |  |  | Possible Formats |  |  |  | Referencing <br> Permitted | Dynamic <br> Definition |  |  |  |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :---: | :---: |
| operand5 | C | S |  |  |  | A |  |  | B |  |  |  | yes | no |

Syntax Element Description:

| ABSOLUTE | This is the default option. It results in an absolute scan of the field <br> for the specified value regardless of what other characters may <br> surround the value. |
| :--- | :--- |
| WITH DELIMITERS | Is used to scan for a value which is delimited by blanks or by any <br> characters that are neither letters nor numeric characters. |
| WITH DELIMITERS <br> operand5 | Is used to scan for a value which is delimited by the character(s) <br> specified in operand5. |

## DELETE REPLACE Clause

```
[AND] { DELETE [FIRST] 
```

Operand Definition Table:

| Operand | Possible <br> Structure |  |  |  |  |  | Possible Formats |  |  |  | Referencing <br> Permitted |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :---: | :---: |
| operand6 | C | S |  |  | A | U |  | B |  |  |  |
| Dynamic <br> Definition |  |  |  |  |  |  |  |  |  |  |  |

Syntax Element Description:

| DELETE | Is used to delete the first (or all) occurrence(s) of the search-value (operand4) in the <br> content of operandl. |
| :--- | :--- |
| REPLACE | Is used to replace the first (or all) occurrence(s) of the search-value (operand4) in <br> operandl by the replace value specified in operand6. |
| FIRST | If you specify the keyword FIRST, only the first identical value will be <br> deleted/replaced. |

## Notes:

1. If the REPLACE operation results in more characters being generated than will fit into operandl, you will receive an error message.
2. If operandl is a dynamic variable, a REPLACE operation may cause its length to be increased or decreased; a DELETE operation may cause its length to be set to zero. The current length of a dynamic variable can be ascertained by using the system variable *LENGTH. For general information on dynamic variables, see Using Dynamic Variables.

## GIVING Clause



Operand Definition Table:

| Operand | Possible <br> Structure |  |  |  |  |  | Possible Formats |  |  |  | Referencing <br> Permitted | Dynamic <br> Definition |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :---: |
| operand7 | S |  |  | N | P | I |  |  |  |  | yes | yes |

Syntax Element Description:

| GIVING | If only the keyword GIVING is specified, this corresponds to GIVING <br> NUMBER (default). |
| :--- | :--- |
| NUMBER | Is used to obtain information on how many times the search value (operand4) <br> was found in the field (operandl) whose content is to be examined. |
| POSITION | Is used to obtain the byte position within operandl (or the substring of <br> operandl) where the first value identical to operand4 was found. |
| LENGTH | Is used to obtain the remaining content length of operandl (or the substring of <br> operandl) after all delete/replace operations have been performed. Trailing <br> blanks are ignored. |
| operand7 | The number of occurrences of the search-value. If the REPLACE FIRST or <br> DELETE FIRST option is also used, the number will not exceed 1. |
| INDEX <br> operand7...3 | See below. |

## GIVING INDEX

```
[GIVING] INDEX [IN] operand7 ... 3
```

This option is only applicable if the underlying field to be examined is an array field.
Syntax Element Description:

| INDEX | GIVING INDEX is used to obtain the array occurrence number (index) of operandl in <br> which the first search-value (operand4) was found. |
| :--- | :--- |
| operand7...3 | operand7 must be specified as many times as there are dimensions in operandl <br> (maximum three times). operand7 will return 0 if the search-value is found in none of <br> the occurrences. |

## Note:

If the index range of operandl includes the occurrence 0 (e.g. $0: 5$ ), a value of 0 in operand 7 is ambiguous. In this case, an additional GIVING NUMBER clause should be used to ascertain whether the search-value was actually found or not.

## Syntax 2 - EXAMINE TRANSLATE

| EXAMINE $\left\{\begin{array}{l}\text { operandl } \\ \text { SUBSTRING } \\ \text { (operand1,operand2,operand3) }\end{array}\right\}$ [AND] |
| ---: | :--- |
| TRANSLATE $\left\{\begin{array}{c}\text { INTO }\left\{\begin{array}{c}\text { UPPER } \\ \text { LOWER }\end{array}\right\} \text { [CASE] } \\ \text { USING [INVERTED] operand } 4\end{array}\right\}$ |

For an explanation of the symbols used in the syntax diagram, see Syntax Symbols.

## Syntax Description - Syntax 2

The EXAMINE TRANSLATE statement is used to translate the characters contained in a field into upper-case or lower-case, or into other characters.

Operand Definition Table:

| Operand | Possible <br> Structure |  |  |  |  |  |  |  | Possible Formats |  |  |  |  | Referencing <br> Permitted | Dynamic <br> Definition |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :---: | :---: |
| operand1 |  | S | A |  | A |  |  |  | B |  |  |  | yes | no |  |
| operand2 | C | S |  |  |  |  | N | P | I | B* |  |  |  | yes | no |
| operand3 | C | S |  |  |  |  | N | P | I | B* |  |  |  | yes | no |
| operand4 |  | S | A |  |  | A |  |  |  | B |  |  |  | yes | no |

*Format B of operand2 and operand3 may be used only with a length of less than or equal to 4. Syntax Element Description:

| EXAMINE operandl | Complete Field Content Translation: <br> operandl is the field whose content is to be translated. |
| :---: | :---: |
| EXAMINE <br> SUBSTRING <br> operand1 operand2 operand3 | Partial Field Content Translation: <br> Normally, the entire content of a field is translated. <br> With the SUBSTRING option, you translate only a certain part of the field. After the field name (operand1) in the SUBSTRING clause, you specify first the starting position (operand2) and then the length (operand3) of the field portion to be examined. <br> For example, to translate the 5th to 12th position inclusive of a field \#A, you would specify: <br> EXAMINE SUBSTRING (\#A, 5, 8) AND TRANSLATE ... <br> Note: <br> If you omit operand2, the starting position is assumed to be 1 . If you omit operand3, the length is assumed to be from the starting position to the end of the field. |
| $\begin{aligned} & \text { TRANSLATE } \\ & \text { INTO UPPER } \\ & \text { CASE } \end{aligned}$ | Upper Case Translation: <br> The content of operand 1 will be translated into upper case. |
| TRANSLATE INTO LOWER CASE | Lower Case Translation: <br> The content of operandl will be translated into lower case. |
| TRANSLATE USING operand4 | Translation Table to be Used: <br> operand4 is the translation table to be used for character translation. The table must be of format/length A2 or B2. <br> Note: <br> If for a character to be translated more than one translation is defined in the translation table, the last of these translations applies. |
| INVERTED | If you specify the keyword INVERTED, the translation table (operand4) will be used inverted; that is, the translation direction will be reversed. |

## Syntax 3 - EXAMINE for Unicode Graphemes

| EXAMINE [FULL [VALUE [OF]]] $\left\{\begin{array}{l}\text { operand1 } \\ \text { SUBSTRING(operand1, operand2, operand3) }\end{array}\right.$ |  |
| :---: | :---: |
| [FOR] | $\left\{\begin{array}{l}\text { CHARPOSITION operand4 CHARLENGTH operand5 } \\ \text { CHARPOSITION operand4 } \\ \text { CHARLENGTH } \text { operand5 }\end{array}\right\}$ |
| [GIVING] POSITION IN ope | d6[[GIVING] LENGTH IN operand7] |

For an explanation of the symbols used in the syntax diagram, see Syntax Symbols.

## Syntax Description - Syntax 3

A "grapheme" is what a user normally thinks of as a character. In most cases, a UTF-16 code unit (= U format character) is a grapheme, however, a grapheme can also consist of several code units. Examples are: a sequence of a base character followed by combining characters or a surrogate pair. For more information on graphemes and other Unicode terms, see The Unicode Standard at http://www.unicode.org/.

The EXAMINE statement for U format operands in general operates on code units. However, with the CHARPOSITION and CHARLENGTH clauses it is possible to obtain the starting position and length (in terms of code units) of a graphemes sequence. The returned code unit values can then be used in other statements/clauses which require code unit operands (e.g. in a SUBSTRING clause).

For further information on this syntax of the EXAMINE statement, see also Unicode and Code Page Support in the Natural Programming Language, section Statements, EXAMINE.

Operand Definition Table:

| Operand | Possible <br> Structure |  |  |  | Possible Formats |  |  |  |  |  |  |  | Referencing Permitted | Dynamic Definition |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| operand1 | C | S | A |  | U |  |  |  | B |  |  |  | yes | no |
| operand 2 | C | S |  |  |  | N | P | I | B* |  |  |  | yes | no |
| operand3 | C | S |  |  |  | N | P | I | B* |  |  |  | yes | no |
| operand4 | C | S |  |  |  | N | P | I |  |  |  |  | yes | no |
| operand5 | C | S |  |  |  | N | P | I |  |  |  |  | yes | no |
| operand6 | C | S |  |  |  | N | P | I |  |  |  |  | yes | no |
| operand7 | C | S |  |  |  | N | P | I |  |  |  |  | yes | no |

* Format B of operand2 and operand3 may be used only with a length of less than or equal to 4 .

Syntax Element Description:
FULL
If FULL is specified for an operand, the entire value, including trailing blanks, will be processed. If FULL is not specified, trailing blanks in the operand will be ignored.
$\left.\begin{array}{|l|l|}\hline \begin{array}{l}\text { SUBSTRING operand1 operand2 } \\ \text { operand3 }\end{array} & \begin{array}{l}\text { Normally, the content of a field is examined from the beginning } \\ \text { of the field to the end or to the last non-blank character. } \\ \text { With the SUBSTRING option, you examine only a certain part } \\ \text { of the field. After the field name (operand1) in the SUBSTRING } \\ \text { clause, you specify first the starting position (operand2) and } \\ \text { then the length (operand3) of the field portion to be examined. } \\ \text { operand2 and operand3 are specified in terms of code units. } \\ \text { For example, to examine the 5th to 12th position inclusive of a } \\ \text { field \#A, you would specify: }\end{array} \\ & \begin{array}{l}\text { EXAMINE sUBSTRING (\#A, 5, 8) }\end{array} \\ \text { Notes: } \\ \text { 1. If you omit operand2, the starting position is assumed to be } \\ \text { 1. } \\ \text { 2. If you omit operand3, the length is assumed to be from the } \\ \text { starting position to the end of the field. } \\ \text { 3. If SUBSTRING is used in conjunction with a DYNAMIC } \\ \text { variable, the field behaves like a fixed length variable; that } \\ \text { is, the length (*LENGTH) does not change as a result of the } \\ \text { EXAMINE operation, regardless of whether a DELETE or } \\ \text { REPLACE operation was executed or not. }\end{array}\right\}$

## Notes:

1. Either the CHARPOSITION or the CHARLENGTH clause or both must be specified.
2. Either the GIVING POSITION or GIVING LENGTH clause or both must be specified.

## Examples

- Example 1 - EXAMINE
- Example 2 - EXAMINE SUBSTRING, PATTERN, TRANSLATE
- Example 3 - EXAMINE TRANSLATE
- Example 4 - EXAMINE for Unicode Graphemes


## Example 1 - EXAMINE

```
** Example 'EXMEX1': EXAMINE
************************************************************************
DEFINE DATA LOCAL
1 #TEXT (A40)
1 #A (A1)
1 #START (N2)
1 #NMB1 (N2)
1 #NMB2 (N2)
1 #NMB3 (N2)
1 #NMBEX2 (N2)
1 #NMBEX3 (N2)
1 #NMBEX4 (N2)
1 #POSEX5 (N2)
1 #LGHEX6 (N2)
1 #NMBEX7 (N2)
1 #NMBEX8 (N2)
END-DEFINE
*
WRITE 'EXAMPLE 1 (GIVING NUMBER, WITH DELIMITER)'
MOVE 'ABC A B C .A. .B. .C. -A- -B- -C- ' TO #TEXT
ASSIGN #A = 'A'
EXAMINE #TEXT FOR #A GIVING NUMBER #NMB1
EXAMINE #TEXT FOR #A WITH DELIMITER GIVING NUMBER #NMB2
EXAMINE #TEXT FOR #A WITH DELIMITER '.' GIVING NUMBER #NMB3
WRITE NOTITLE '=' #NMB1 '=' #NMB2 '=' #NMB3
*
WRITE / 'EXAMPLE 2 (WITH DELIMITER, REPLACE, GIVING NUMBER)'
WRITE '=' #TEXT
EXAMINE #TEXT FOR #A WITH DELIMITER '-' REPLACE WITH '*'
                GIVING NUMBER #NMBEX2
WRITE '=' #TEXT '=' #NMBEX2
*
WRITE / 'EXAMPLE 3 (REPLACE, GIVING NUMBER)'
WRITE '=' #TEXT
EXAMINE #TEXT ' ' REPLACE WITH '+' GIVING NUMBER #NMBEX3
WRITE '=' #TEXT '=' #NMBEX3
*
WRITE / 'EXAMPLE 4 (FULL, REPLACE, GIVING NUMBER)'
WRITE '=' #TEXT
EXAMINE FULL #TEXT ' ' REPLACE WITH '+' GIVING NUMBER #NMBEX4
WRITE '=' #TEXT '=' #NMBEX4
*
```

```
WRITE / 'EXAMPLE 5 (DELETE, GIVING POSITION)'
WRITE '=' #TEXT
EXAMINE #TEXT '+' DELETE GIVING POSITION #POSEX5
WRITE '=' #TEXT '=' #POSEX5
*
WRITE / 'EXAMPLE 6 (DELETE, GIVING LENGTH)'
WRITE '=' #TEXT
EXAMINE #TEXT FOR 'A' DELETE GIVING LENGTH #LGHEX6
WRITE '=' #TEXT '=' #LGHEX6
*
*
NEWPAGE
*
MOVE 'ABC A B C .A. .B. .C. -A- -B- -C- ' TO #TEXT
*
ASSIGN #A = 'A B C'
ASSIGN #START = 6
*
WRITE / 'EXAMPLE 7 (SUBSTRING, GIVING NUMBER)'
WRITE '=' #TEXT
EXAMINE SUBSTRING(#TEXT,#START,9) FOR #A GIVING NUMBER #NMBEX7
WRITE '=' #TEXT '=' #NMBEX7
*
WRITE / 'EXAMPLE 8 (PATTERN, GIVING NUMBER)'
WRITE '=' #TEXT
EXAMINE #TEXT FOR PATTERN '-A-' GIVING NUMBER #NMBEX8
WRITE '=' #TEXT '=' #NMBEX8
*
END
```


## Output of Program EXMEX1:



## Example 2 - EXAMINE SUBSTRING, PATTERN, TRANSLATE

```
** Example 'EXMEX2': EXAMINE TRANSLATE
************************************************************************
DEFINE DATA LOCAL
1 #TEXT (A50)
1 #TAB (A2/1:10)
1 #START (N2)
END-DEFINE
*
MOVE 'ABC A B C .A. .B. .C. -A- -B- -C- ' TO #TEXT
*
MOVE 'AX' TO #TAB(1)
MOVE 'BY' TO #TAB(2)
MOVE 'CZ' TO #TAB(3)
*
*
WRITE 'EXAMPLE 1 (USING TRANSLATION TABLE)'
WRITE '=' #TEXT
EXAMINE #TEXT TRANSLATE USING #TAB(*)
WRITE NOTITLE '=' #TEXT
*
WRITE / 'EXAMPLE 2 (USING INVERTED TRANSLATION TABLE)'
WRITE '=' #TEXT
EXAMINE #TEXT TRANSLATE USING INVERTED #TAB(*)
WRITE NOTITLE '=' #TEXT
*
WRITE / 'EXAMPLE 3 (USING SUBSTRING, LOWER CASE)'
WRITE '=' #TEXT
ASSIGN #START = 13
EXAMINE SUBSTRING(#TEXT,#START,15) TRANSLATE INTO LOWER CASE
WRITE '=' #TEXT
END
```


## Output of Program EXMEX2:

```
EXAMPLE 1 (USING TRANSLATION TABLE)
#TEXT: ABC A B C .A. .B. .C. -A- -B- -C-
#TEXT: XYZ X Y Z .X. .Y. .Z. -X- -Y- -Z-
EXAMPLE 2 (USING INVERTED TRANSLATION TABLE)
#TEXT: XYZ X Y Z .X. .Y. .Z. -X- -Y- -Z-
#TEXT: ABC A B C .A. .B. .C. -A- -B- -C-
EXAMPLE 3 (USING SUBSTRING, LOWER CASE)
#TEXT: ABC A B C .A. .B. .C. -A- -B- -C-
#TEXT: ABC A B C .a. .b. .c. -A- -B- -C-
```


## Example 3 - EXAMINE TRANSLATE

```
** Example 'EXMEX2': EXAMINE TRANSLATE
************************************************************************
DEFINE DATA LOCAL
1 #TEXT (A50)
1 #TAB (A2/1:10)
1 #START (N2)
END-DEFINE
*
MOVE 'ABC A B C .A. .B. .C. -A- -B- -C- ' TO #TEXT
*
MOVE 'AX' TO #TAB(1)
```

```
MOVE 'BY' TO #TAB(2)
MOVE 'CZ' TO #TAB(3)
*
*
WRITE 'EXAMPLE 1 (USING TRANSLATION TABLE)'
WRITE '=' #TEXT
EXAMINE #TEXT TRANSLATE USING #TAB(*)
WRITE NOTITLE '=' #TEXT
*
WRITE / 'EXAMPLE 2 (USING INVERTED TRANSLATION TABLE)'
WRITE '=' #TEXT
EXAMINE #TEXT TRANSLATE USING INVERTED #TAB(*)
WRITE NOTITLE '=' #TEXT
*
WRITE / 'EXAMPLE 3 (USING SUBSTRING, LOWER CASE)'
WRITE '=' #TEXT
ASSIGN #START = 13
EXAMINE SUBSTRING(#TEXT,#START,15) TRANSLATE INTO LOWER CASE
WRITE '=' #TEXT
END
```


## Output of Program EXMEX2:

```
EXAMPLE 1 (USING TRANSLATION TABLE)
#TEXT: ABC A B C .A. .B. .C. -A- -B- -C-
#TEXT: XYZ X Y Z .X. .Y. .Z. -X- -Y- -Z-
EXAMPLE 2 (USING INVERTED TRANSLATION TABLE)
#TEXT: XYZ X Y Z .X. .Y. .Z. -X- -Y- -Z-
#TEXT: ABC A B C .A. .B. .C. -A- -B- -C-
EXAMPLE 3 (USING SUBSTRING, LOWER CASE)
\#TEXT: ABC A B C .A. .B. .C. \(-\mathrm{A}-\quad-\mathrm{B}-\quad-\mathrm{C}-\)
```


## Example 4 - EXAMINE for Unicode Graphemes

This example demonstrates the analysis of a Unicode string containg the characters ä und ü. Both characters are defined as base character followed by a combining character: ä is coded with $U+0061$ followed by $\mathrm{U}+0308$, and $\ddot{\mathrm{u}}$ is coded with $\mathrm{U}+0075$ followed by $\mathrm{U}+0308$.

```
DEFINE DATA LOCAL
1 #U (U20)
1 #START (I2)
1 #POS (I2)
1 #LEN (I2)
END-DEFINE
#U := U'AB'-UH'00610308'-U'CD' -UH'00750308'-U'EF'
*
REPEAT
    #START := #START + 1
    EXAMINE #U FOR CHARPOSITION #START
                                    CHARLENGTH 1
                                    GIVING POSITION IN #POS
                                    LENGTH IN #LEN
*
    INPUT (AD=O) MARK POSITION #POS IN FIELD *#U
        ' UNICODE-STRING:' #U (AD=MI)
    // ' CHARACTER NO.:' #START (EM=9)
    / 'STARTS AT BYTE POSITION:' #POS (EM=9)
```

```
    / , AND THE LENGTH IS:' #LEN (EM=9)
WHILE #POS NE O
END-REPEAT
END
```

Output:

| Mainframe Environments: | Windows, UNIX and OpenVMS Environments (with Natural Web I/O Interface): |
| :---: | :---: |
| ```UNICODE-STRING: ABa?CDu?EE \\ ChARACTER NO.: 1None``` | UNICODE-STRING: ABäCDüEF <br> Character no.: 1 <br> STARTS AT BYTE POSITION: 1 <br> AND THE LENGTH IS: 1 |
| Press ENTER to continue. | Press ENTER to continue. |
|  | UNICODE-STRING: ABäCDÜEF CHARACTER NO.: 2 STARTS AT BYTE POSITION: 2 AND THE LENGTH IS: 1 |
| Press ENTER to continue. | Press ENTER to continue. |
| Note that the character in position 3 is a combining character sequence and is two code units long. |  |
| UNICODE-STRING: AB<b>a</b>?CDu?E <br> CHARACTER NO.: 3 <br> STARTS AT BYTE POSITION: 3 <br> AND THE LENGTH IS: 2 | UNICODE-STRING: AB<b>ä</b>CDüEF CHARACTER NO.: 3 STARTS AT BYTE POSITIN: 3 AND THE LENGTH IS: 2 |
| And so on. | And so on. |

