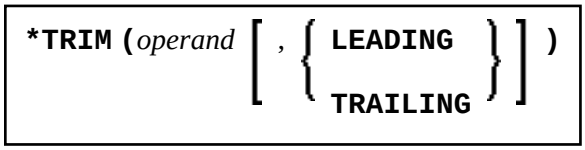
*TRIM - Entfernen von führenden und/oder nachfolgenden Leerstellen



Format/Länge: wie bei *operand* (A oder B)/DYNAMIC.

Die folgenden Themen werden behandelt:

- Funktion
- Einschränkungen
- Syntax-Beschreibung
- Beispiele

Funktion

Die Systemfunktion *TRIM dient zum Entfernen von führenden und/oder nachfolgenden Leerstellen aus einer aphanumerischen oder binären Zeichenkette. Der Inhalt von *operand* bleibt dabei unverändert. Bei Verwendung einer dynamischen Variablen als *operand*, wird die Länge dieser Variablen dem Ergebnis entsprechend angepasst.

Die Systemfunktion *TRIM kann als *operand* an jeder Stelle eines Statements angegeben werden, an der ein Operand mit Format A oder B zulässig ist.

Einschränkungen

Für die Verwendung der Systemfunktion *TRIM gelten folgende Einschränkungen:

- *TRIM darf nicht an Stellen verwendet werden, an denen eine Zielvariable erwartet wird.
- *TRIM darf nicht in einer anderen Systemfunktion verschachtelt werden.
- Wenn der Operand eine statische Variable ist, kann man mit *TRIM keine führenden Leerstellen entfernen, weil bei statischen Variablen die verbleibenden nachfolgenden Stellen des Variablenspeichers mit Leerzeichen aufgefüllt werden.

Syntax-Beschreibung

Operanden-Definitionstabelle:

Operand	Mögliche Struktur					Mögliche Formate											Referenzierung erlaubt	Dynam. Definition
operand	C	S	A			A	U	B									ja	nein

Syntax-Elementbeschreibung:

`*TRIM(`*`operand`*`, LEADING)`	**Führende Leerstellen entfernen** Bei Angabe des Schlüsselworts `LEADING` als zweites Argument werden alle führenden Leerstellen aus dem in *`operand`* enthaltenen String entfernt.
`*TRIM(`*`operand`*`, TRAILING)`	**Nachfolgende Leerstellen entfernen** Bei Angabe des Schlüsselworts `TRAILING` als zweites Argument werden alle nachfolgenden Leerstellen aus dem in *`operand`* enthaltenen String entfernt.
`*TRIM(`*`operand`*`)`	**Führende und Nachfolgende Leerstellen entfernen** Wenn kein zweites Schlüsselwort als Argument angegeben wird, bewirkt`TRIM`, dass alle führenden und nachfolgenden Leerstellen aus dem in *`operand`* enthaltenen String entfernt werden.

Beispiele

- Beispiel 1 - Verwendung eines alphanumerischen Arguments
- Beispiel 2 - Verwendung eines binären Arguments

Beispiel 1 - Verwendung eines alphanumerischen Arguments

```
DEFINE DATA LOCAL
/*****************************
/* STATIC VARIABLE DEFINITIONS
/*****************************
1 #SRC  (A15) INIT <'  ab  CD  '>
1 #DEST (A15)

/* FOR PRINT OUT WITH DELIMITERS
1 #SRC-PRN  (A20)
1 #DEST-PRN (A20)

/*****************************
/* DYNAMIC VARIABLE DEFINITIONS
/*****************************
1 #DYN-SRC (A)DYNAMIC INIT <'  ab  CD  '>
1 #DYN-DEST (A)DYNAMIC

/* FOR PRINT OUT WITH DELIMITERS
1 #DYN-SRC-PRN  (A)DYNAMIC
1 #DYN-DEST-PRN (A)DYNAMIC

END-DEFINE
```

```
PRINT 'static variable definition:'
PRINT '---------------------------'
COMPRESS FULL ':' #SRC ':' TO #SRC-PRN LEAVING NO SPACE
PRINT ' '
PRINT ' 123456789012345      123456789012345'

MOVE *TRIM(#SRC, LEADING) TO #DEST
COMPRESS FULL ':' #DEST ':' TO #DEST-PRN LEAVING NO SPACE
DISPLAY #SRC-PRN #DEST-PRN '*TRIM(#SRC, LEADING)'

MOVE *TRIM(#SRC, TRAILING) TO #DEST
COMPRESS FULL ':' #DEST ':' TO #DEST-PRN LEAVING NO SPACE
DISPLAY #SRC-PRN #DEST-PRN '*TRIM(#SRC, TRAILING)'

MOVE *TRIM(#SRC) TO #DEST
COMPRESS FULL ':' #DEST ':' TO #DEST-PRN LEAVING NO SPACE
DISPLAY #SRC-PRN #DEST-PRN '*TRIM(#SRC)'

PRINT ' '
PRINT 'dynamic variable definition:'
PRINT '----------------------------'
COMPRESS FULL ':' #DYN-SRC ':' TO #DYN-SRC-PRN LEAVING NO SPACE
PRINT ' '
PRINT ' 1234567890           12345678'

MOVE *TRIM(#DYN-SRC, LEADING) TO #DYN-DEST
COMPRESS FULL ':' #DYN-DEST ':' TO #DYN-DEST-PRN LEAVING NO SPACE
DISPLAY (AL=20) #DYN-SRC-PRN #DYN-DEST-PRN '*TRIM(#SRC, LEADING)'

MOVE *TRIM(#DYN-SRC, TRAILING) TO #DYN-DEST
COMPRESS FULL ':' #DYN-DEST ':' TO #DYN-DEST-PRN LEAVING NO SPACE
DISPLAY (AL=20) #DYN-SRC-PRN #DYN-DEST-PRN '*TRIM(#SRC, TRAILING)'

MOVE *TRIM(#DYN-SRC) TO #DYN-DEST
COMPRESS FULL ':' #DYN-DEST ':' TO #DYN-DEST-PRN LEAVING NO SPACE
DISPLAY (AL=20) #DYN-SRC-PRN #DYN-DEST-PRN '*TRIM(#SRC)'

PRINT ' '
PRINT '":" := delimiter character to show the start and ending of a string!'
END
```

Ausgabe:

```
      #SRC-PRN             #DEST-PRN
-------------------- ----------------------

static variable definition:
---------------------------

 123456789012345         123456789012345
:  ab  CD       :       :ab  CD         :     *TRIM(#SRC, LEADING)
:  ab  CD       :       :  ab  CD       :     *TRIM(#SRC, TRAILING)
:  ab  CD       :       :ab  CD         :     *TRIM(#SRC)

dynamic variable definition:
----------------------------

 1234567890              12345678
:  ab  CD  :            :ab  CD  :            *TRIM(#SRC, LEADING)
```

```
:  ab  CD  :            :  ab  CD:            *TRIM(#SRC, TRAILING)
:  ab  CD  :            :ab  CD:              *TRIM(#SRC)

':' := delimiter character to show the start and ending of a string!
```

Beispiel 2 - Verwendung eines binären Arguments

```
DEFINE DATA LOCAL
/*****************************
/* STATIC VARIABLE DEFINITIONS
/*****************************
1 #SRC  (B10) INIT <H'2020FFFF2020FFFF2020'>
1 #DEST (B10)

/*****************************
/* DYNAMIC VARIABLE DEFINITIONS
/*****************************
1 #DYN-SRC (B)DYNAMIC INIT <H'2020FFFF2020FFFF2020'>
1 #DYN-DEST (B)DYNAMIC
END-DEFINE

FORMAT LS=100

PRINT 'static variable definition:
PRINT '--------------------------'
MOVE *TRIM(#SRC, LEADING) TO #DEST
PRINT #SRC #DEST '*TRIM(#SRC, LEADING)'

MOVE *TRIM(#SRC, TRAILING) TO #DEST
PRINT #SRC #DEST '*TRIM(#SRC, TRAILING)'

MOVE *TRIM(#SRC) TO #DEST
PRINT #SRC #DEST '*TRIM(#SRC)'

PRINT ' '
PRINT 'dynamic variable definition:'
PRINT '----------------------------'

MOVE *TRIM(#DYN-SRC, LEADING) TO #DYN-DEST
PRINT #DYN-SRC #DYN-DEST '    *TRIM(#SRC, LEADING)'

MOVE *TRIM(#DYN-SRC, TRAILING) TO #DYN-DEST
PRINT #DYN-SRC #DYN-DEST '    *TRIM(#SRC, TRAILING)'

MOVE *TRIM(#DYN-SRC) TO #DYN-DEST
PRINT #DYN-SRC #DYN-DEST '        *TRIM(#SRC)'

PRINT ' '

PRINT 'hex."20" := space character'
END
```

Ausgabe:

```
static variable definition:
---------------------------

2020FFFF2020FFFF2020 0000FFFF2020FFFF2020     *TRIM(#src, leading)
2020FFFF2020FFFF2020 00002020FFFF2020FFFF     *TRIM(#src, trailing)
2020FFFF2020FFFF2020 00000000FFFF2020FFFF     *TRIM(#src)

dynamic variable definition:
```

```
---------------------------

2020FFFF2020FFFF2020 FFFF2020FFFF2020         *TRIM(#src, leading)
2020FFFF2020FFFF2020 2020FFFF2020FFFF         *TRIM(#src, trailing)
2020FFFF2020FFFF2020 FFFF2020FFFF             *TRIM(#src)

hex.’20’ := space character
```