# COBOL - STRING HANDLING

 

String handling statements in COBOL are used to do multiple functional operations on strings. Following are the string handling statements:

- Inspect
- String
- Unstring

## Inspect

Inspect verb is used to count or replace the characters in a string. String operations can be performed on alphanumeric, numeric, or alphabetic values. Inspect operations are performed from left to right. The options used for the string operations are as follows:

## Tallying

Tallying option is used to count the string characters.

### Syntax

Following is the syntax of Tallying option:

```
INSPECT input-string
TALLYING output-count FOR ALL CHARACTERS
```

### The parameters used are:

- input-string : The string whose characters are to be counted.
- output-count : Data item to hold the count of characters.

### Example

```
IDENTIFICATION DIVISION.
PROGRAM-ID. HELLO.

DATA DIVISION.
   WORKING-STORAGE SECTION.
   01 WS-CNT1 PIC 9(2) VALUE 0.
   01 WS-CNT2 PIC 9(2) VALUE 0.
   01 WS-STRING PIC X(15) VALUE 'ABCDACDADEAAAFF'.

PROCEDURE DIVISION.
   INSPECT WS-STRING TALLYING WS-CNT1 FOR ALL CHARACTERS.
   DISPLAY "WS-CNT1 : "WS-CNT1.
   INSPECT WS-STRING TALLYING WS-CNT2 FOR ALL 'A'.
   DISPLAY "WS-CNT2 : "WS-CNT2

STOP RUN.
```

**JCL** to execute the above COBOL program.

```
//SAMPLE JOB(TESTJCL,XXXXXX),CLASS=A,MSGCLASS=C
//STEP1 EXEC PGM=HELLO
```

When you compile and execute the above program, it produces the following result:

```
WS-CNT1 : 15
WS-CNT2 : 06
```

## Replacing

Replacing option is used to replace the string characters.

### Syntax

Following is the syntax of Replacing option:

```
INSPECT input-string REPLACING ALL char1 BY char2.
```

### The parameter used is:

**input-string** : The string whose characters are to be replaced from char1 to char2.

### Example

```
IDENTIFICATION DIVISION.
PROGRAM-ID. HELLO.

DATA DIVISION.
   WORKING-STORAGE SECTION.
   01 WS-STRING PIC X(15) VALUE 'ABCDACDADEAAAFF'.

PROCEDURE DIVISION.
   DISPLAY "OLD STRING : "WS-STRING.
   INSPECT WS-STRING REPLACING ALL 'A' BY 'X'.
   DISPLAY "NEW STRING : "WS-STRING.

STOP RUN.
```

**JCL** to execute the above COBOL program.

```
//SAMPLE JOB(TESTJCL,XXXXXX),CLASS=A,MSGCLASS=C
//STEP1 EXEC PGM=HELLO
```

When you compile and execute the above program, it produces the following result:

```
OLD STRING : ABCDACDADEAAAFF
NEW STRING : XBCDXCDXDEXXXFF
```

## String

String verb is used to concatenate the strings. Using STRING statement, two or more strings of characters can be combined to form a longer string. 'Delimited By' clause is compulsory.

### Syntax

Following is the syntax of String verb:

```
STRING ws-string1 DELIMITED BY SPACE
   ws-string2 DELIMITED BY SIZE
   INTO ws-destination-string
   WITH POINTER ws-count
   ON OVERFLOW DISPLAY message1
   NOT ON OVERFLOW DISPLAY message2
END-STRING.
```

### Following are the details of the used parameters:

- ws-string1 and ws-string2 : Input strings to be concatenated
- ws-string : Output string
- ws-count : Used to count the length of new concatenated string

- Delimited specifies the end of string
- Pointer and Overflow are optional

**Example**

```
IDENTIFICATION DIVISION.
PROGRAM-ID. HELLO.

DATA DIVISION.
   WORKING-STORAGE SECTION.
   01 WS-STRING PIC A(30).
   01 WS-STR1 PIC A(15) VALUE 'Tutorialspoint'.
   01 WS-STR2 PIC A(7) VALUE 'Welcome'.
   01 WS-STR3 PIC A(7) VALUE 'To AND'.
   01 WS-COUNT PIC 99 VALUE 1.

PROCEDURE DIVISION.
   STRING WS-STR2 DELIMITED BY SIZE
      WS-STR3 DELIMITED BY SPACE
      WS-STR1 DELIMITED BY SIZE
      INTO WS-STRING
      WITH POINTER WS-COUNT
      ON OVERFLOW DISPLAY 'OVERFLOW!'
   END-STRING.

   DISPLAY 'WS-STRING : 'WS-STRING.
   DISPLAY 'WS-COUNT : 'WS-COUNT.

STOP RUN.
```

**JCL** to execute the above COBOL program:

```
//SAMPLE JOB(TESTJCL,XXXXXX),CLASS=A,MSGCLASS=C
//STEP1 EXEC PGM=HELLO
```

When you compile and execute the above program, it produces the following result:

```
WS-STRING : WelcomeToTutorialspoint
WS-COUNT : 25
```

## Unstring

Unstring verb is used to split one string into multiple sub-strings. Delimited By clause is compulsory.

**Syntax**

Following is the syntax of Unstring verb:

```
UNSTRING ws-string DELIMITED BY SPACE
INTO ws-str1, ws-str2
WITH POINTER ws-count
ON OVERFLOW DISPLAY message
NOT ON OVERFLOW DISPLAY message
END-UNSTRING.
```

**Example**

```
IDENTIFICATION DIVISION.
PROGRAM-ID. HELLO.

DATA DIVISION.
   WORKING-STORAGE SECTION.
   01 WS-STRING PIC A(30) VALUE 'WELCOME TO TUTORIALSPOINT'.
   01 WS-STR1 PIC A(7).
```

```
   01 WS-STR2 PIC A(2).
   01 WS-STR3 PIC A(15).
   01 WS-COUNT PIC 99 VALUE 1.

PROCEDURE DIVISION.
   UNSTRING WS-STRING DELIMITED BY SPACE
      INTO WS-STR1, WS-STR2, WS-STR3
   END-UNSTRING.

   DISPLAY 'WS-STR1 : 'WS-STR1.
   DISPLAY 'WS-STR2 : 'WS-STR2.
   DISPLAY 'WS-STR3 : 'WS-STR3.

STOP RUN.
```

**JCL** to execute the above COBOL program:

```
//SAMPLE JOB(TESTJCL,XXXXXX),CLASS=A,MSGCLASS=C
//STEP1 EXEC PGM=HELLO
```

When you compile and execute the above program, it produces the following result:

```
WS-STR1 : WELCOME
WS-STR2 : TO
WS-STR3 : TUTORIALSPOINT
```