RPG

allacciamo le stringhe

Marco Riva

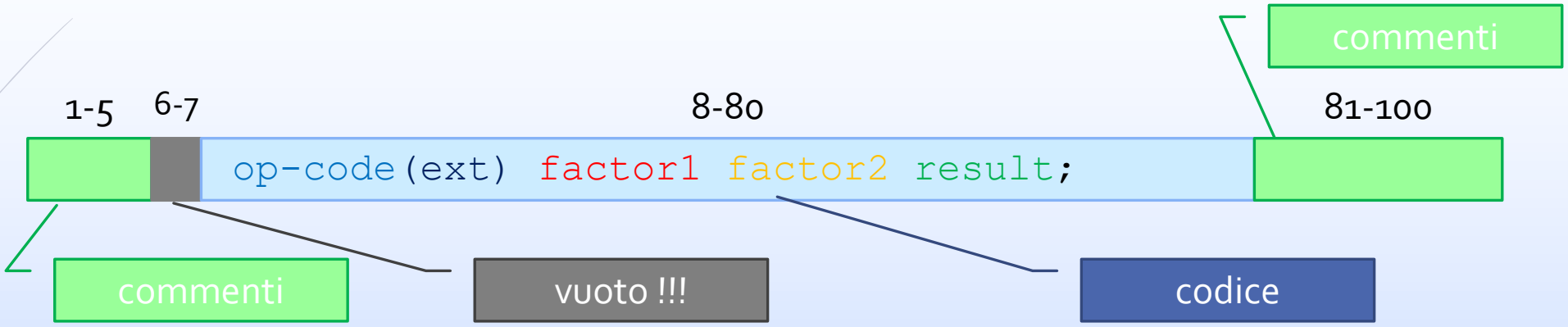
MK1

www.markonetools.it

IBMCHAMPION 2021 Power Systems area

Ultimo aggiornamento: 25/01/2021

Regole base



- ogni istruzione deve terminare con il carattere **;**
- `/free … /end-free` **non** sono più necessari da 7.1 TR7
- si può scrivere un'**istruzione su più righe** senza nessun carattere di continuazione, perché la fine dell'istruzione è determinato dalla presenza del **;**

Codici operativi "obsoleti"

- ATTENZIONE: non tutti i codici operativi possono essere scritti in formato libero: ADD, ANDxx, CABxx, CALL, CALLB, CASxx, **CAT**, **CHECK**, **CHECKR**, COMP, DEFINE, DIV, DOUxx, DOWxx, GOTO, IFxx, MOVE, MOVEA, MOVEL, MULT, MVR, ORxx, PARM, PLIST, **SCAN**, SETOFF, SETON, SQRT, SUB, **SUBST**, TAG, TESTB, TESTN, TESTZ, TIME, WHENxx, **XLATE**, Z-ADD, Z-SUB
- I codici operativi non gestiti dal formato libero possono essere definiti "obsoleti" e "deprecabili"

come manipolare le stringhe in formato libero?

Carattere di continuazione per stringhe

- Simbolo **-**: la stringa continua con la **prima colonna** della riga successiva
- Simbolo **+**: la stringa continua con la **prima colonna non blank** della riga successiva

```
dcl-s Stringa char(40);

Stringa = '*** INIZIO PROGRAMMA ***';
dsply stringa;
Stringa = 'Prima riga -
     seconda riga (segno -)';
dsply stringa;
Stringa = 'Prima riga +
     seconda riga (segno +)';
dsply stringa;

*inlr = *on;
```

```
DSPLY  *** INIZIO PROGRAMMA ***
DSPLY  Prima riga      seconda riga (segno -)
DSPLY  Prima riga seconda riga (segno +)
```

Concatenazione

- L'operatore di concatenazione di variabili e costanti alfanumeriche è **+** e sostituisce l'obsoleto CAT
- Con variabili alfanumeriche a lunghezza fissa non bisogna dimenticare le funzioni **%trim**/**%trimr**/**%triml** per concatenare eliminando i blanks

```
dcl-s sqlstm char(200);

sqlstm = 'select empno from employee';
if ws_Ordinamento = 'Nome';
  sqlstm = %trimr(sqlstm) + ' order by +
           firstnme;';
endif;
```

operatore di concatenazione

carattere di continuazione

Semplificare la concatenazione

Concatenare apici, blank… evitando «mal di testa»

```
dcl-s sqlstm char(100);
dcl-s ws_codice char(6) inz('000010');
dcl-c AP const('''');
```

```
sqlstm = 'select lastname from employee +
                  where empno = ' + '''' + ws_codice + '''';
```

=

```
sqlstm = 'select lastname from employee +
                    where empno = ''' + ws_codice + '''';
```

=

```
sqlstm = 'select lastname from employee +
                      where empno = ' + AP + ws_codice + AP;
```

Semplificare la concatenazione

Concatenare apici, blank… evitando «mal di testa»

```
dcl-s sqlstm char(100);
dcl-c AP const('''');
dcl-c BL const(''' ''');
```

```
sqlstm = 'select lastname from employee +
                    where empno = '' ''';
```

=

```
sqlstm = 'select lastname from employee +
                      where empno = ' + AP + ' ' + AP;
```

=

```
sqlstm = 'select lastname from employee +
                      where empno = ' + BL;
```

Concatenare campi numerici

Per concatenare campi numerici a costanti o variabili alfanumeriche si può ricorrere alla funzione **%char**

```
dcl-s Msg char(100);
dcl-s Nrek int(5);

Msg = 'Sono stati cancellati ' + %char(Nrek) + ' record.';
```

```
dcl-s Msg char(100);
dcl-s Importo packed(13:2);

Msg = 'Valore dell''ordine è ' + %char(Importo) + ' euro.';
```

Punto e a capo

Per inserire un «ritorno a capo» all'interno di una stringa si può concatenare il codice esadecimale **x'25'**

```
dcl-s BodyEmail varchar(500);

BodyEmail = 'In allegato copia della fattura.' + x'25' +
            'Distinti saluti.'
```

=

```
dcl-s BodyEmail varchar(500);
dcl-c LF const(x'25');

BodyEmail = 'In allegato copia della fattura.' + LF +
            'Distinti saluti.'
```

Costanti di uso comune

- Negli esempi precedenti abbiamo usato alcune costanti che possono tornare comode in molti contesti
- E' opportuno riunirle in un sorgente da importare dove serve con un /copy

```
dcl-c xNull const(x'00');  // Null terminator
dcl-c xLF const(x'25');    // Line feed
dcl-c xCR const(x'0D');    // Carriage Return
dcl-c xCRLF const(x'0D25');// Carr.Rtn/Line feed
dcl-c xSpace const(x'40'); // blank
dcl-c xTab const(x'05');   // Tab
dcl-c AP const('''');      // apice
dcl-c BL const(''' ''');   // blank tra apici
```

Sottostringare

- Il codice operativo SUBST viene rimpiazzato dalla funzione **%subst**

```
NomeFile = %subst(Percorso : pos1+1 : pos2-1);
%subst(MioCampo:3:2) = 'AA';
```

Trova

Il codice operativo SCAN è sostituito dalla funzione **%scan** o **%scanr** (scan reverse)

```
Percorso = '/home/qpgmr/123.txt';
UltSlash = %scanr('/':Percorso);
if UltSlash = *zeros;
  NomeFile = Percorso;
else;
  NomeFile = %subst(Percorso:UltSlash+1);
endif;
```

Trova e sostituisci

C'è di più! Con la funzione **%scanrpl** si può cercare una stringa e sostituirla con un'altra

```
%scanrpl(scan string:replacement:source {:scan start {:scan length  })
```

Altre funzioni

- Invece di CHECK o CHECKR si usa la funzione **`%check`** o **`%checkr`**
- Invece di XLATE si usa la funzione **`%xlate`**

Riferimenti

- E-mail aziendale: mriva@sirio-is.it
- Blog: www.markonetools.it
- E-mail blog: info@markonetools.it
- Linkedin: www.linkedin.com/in/marcoriva-mk1
- Twitter: @MarcoRiva73
- Facebook: https://www.facebook.com/markonetools/
- YouTube: https://www.youtube.com/channel/UCb47YJQJCzU-5x4nnGzDu-w