Einführung in Programmierung mit ABAP:

Teil 2: Datentypen, Zeitberechnung, kaufmännisches Rechnen, logische Operationen

Prof. Dr. Peter Hohmann

Technische Hochschule Mittelhessen

FB MNI

www.prof-dr-hohmann.de

Version 2017

Datentypen

- elementare Datentypen
  - feste Länge (c,d,f,i,n,p,t,x)
  - variable Länge (Strings)
- Referenztypen
  - Datenreferenzen
  - Objektreferenzen
- komplexe Datentypen
  - strukturierte Typen (Datenstrukturen)
  - Tabellentypen (interne Tabellen)

eingebaute, elementare Datentypen

Typ	Beschreibung	Byte	Initialwert
f	Float, Gleitpunktzahl	**8**	**0**
i	Integer, ganze Zahl	**4**	**0**
n	numerischer Text	**1 (1-65535)**	**'0....0'**
p	Packed, gepackte Zahl	**8 (1-16)**	**0**
c	Character, alphanum. Text	**1**	**' '**
d	Date, Datumsangabe	**8**	**'00000000'**
t	Time, Zeitangabe	**6**	**'000000'**
x	Hexadezimal	**1 (1-65535)**	**'0....0'**
string / xstring	beliebige Zeichen- / Bytefolgen		

lokale Datendeklarationen

```
DATA  vorname(20)     TYPE c .
DATA  plz(5)          TYPE n .
DATA  wert            TYPE i .
DATA  gehalt(6)       TYPE p DECIMALS 2 .
DATA  zulage          TYPE f VALUE 100 .
```

Wertzuweisung, arithmetische Operationen:

```
vorname = 'Helmut' .
plz     = 53779 .
wert    = 24 .
gehalt  = '2500.60' + zulage .
wert    = wert + 20 .
```

lokale Typ- und Datendeklarationen

```
TYPES   vorname(20)      TYPE c .
TYPES   plz(5)           TYPE n .
TYPES   alter            TYPE i .
TYPES   gehalt(6)        TYPE p DECIMALS 2 .

TYPES:   vorname(20)      TYPE c ,
         plz(5)           TYPE n ,
         alter            TYPE i ,
         gehalt(6)        TYPE p DECIMALS 2 .

DATA    vname            TYPE vorname .
DATA:   plz_code         TYPE plz ,
        kd_alter         TYPE alter ,
        m_gehalt         TYPE gehalt .
```

elementare Datentypen / Datenobjekte

lokale Typdeklaration

```
TYPES typ_name[(laenge)]
   {TYPE typ | LIKE name} [DECIMALS dezimal].
```

Datendeklaration

```
* Typen i, f, d, t, string, xstring
DATA name {TYPE typ | LIKE name}
      [VALUE wert].

* generische Typen c, n, p, x
DATA name[(laenge)] {TYPE typ | LIKE name}
      [DECIMALS dezimal] [VALUE wert].
```

Konstanten und Literale

Deklaration

```
CONSTANTS name[(len)] {TYPE typ | LIKE name}
      [DECIMALS dezimal] VALUE wert .
```

Literale

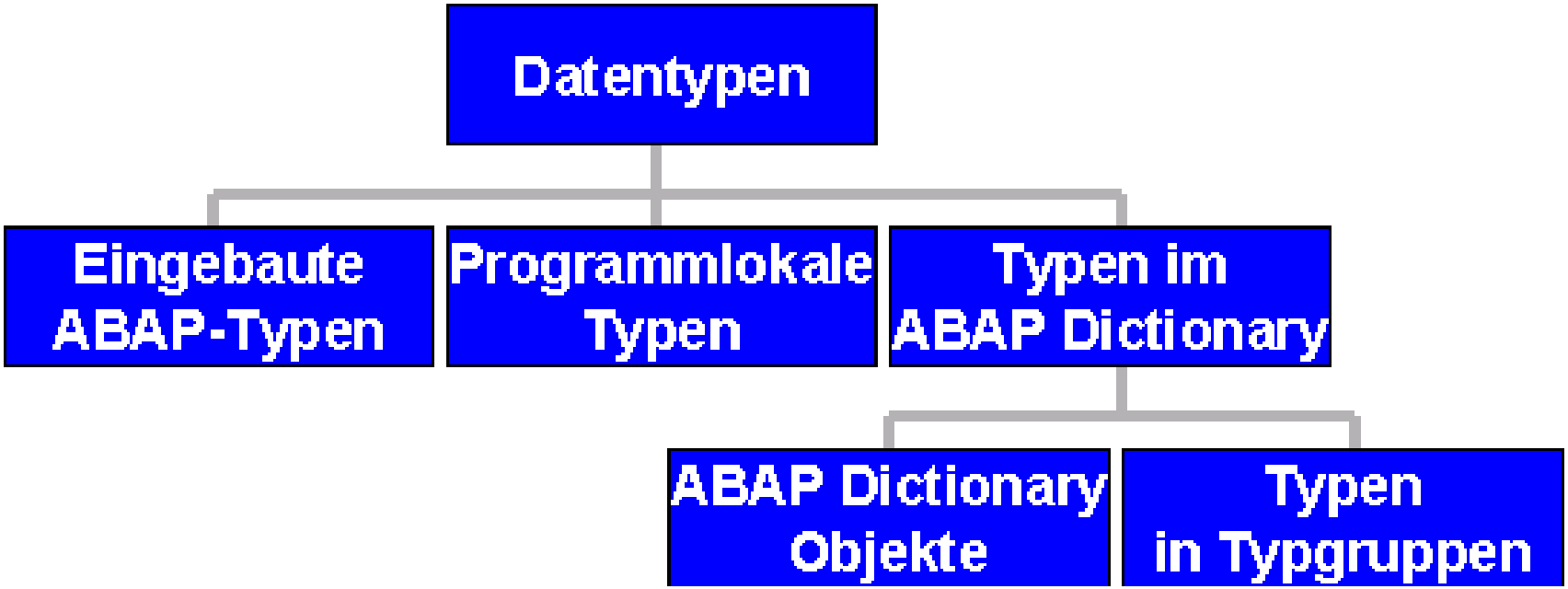
```
'Dies ist ein Textliteral'

1234567890     -4711

'12.34567'   '-4711.123'    '12.34567-'

'12.34567E-04'   '+47.12E+4'     '1E160'
```

Datentypen im SAP-System



Namenskonventionen

- max. 30 Zeichen
- alle Buchstaben (ohne Umlaute), Ziffern und Unterstrich _
- Name muss mit einem Buchstaben beginnen
- nicht erlaubt sind:
  - vordefinierte Namen (z.Bsp. space, sy-uname etc.)
  - ABAP-Schlüsselwörter oder deren Zusätze
- Bindestrich - nicht verwenden, wird beim Zugriff auf Strukturelemente benötigt

Wertzuweisung

```
destination = source .
MOVE source TO destination .
MOVE-CORRESPONDING s_struct TO d_struct .
CLEAR source .
```

Typkonvertierungsregeln

- 1 zu 1 Übertragung bei kompatiblen Datenobjekten
- sinnvolle Typkonvertierung bei konvertiblen Datenobjekten
- Compiler- bzw. Laufzeitfehler bei inkonvertiblen Datenobjekten

arithmetische Berechnungen

Operation	arithm. Ausdruck	spez. Schlüsselwort
Addition	`r = op1 + op2.`	`ADD n TO m.`
Subtraktion	`r = op1 - op2.`	`SUBTRACT n FROM m.`
Multiplikation	`r = op1 * op2.`	`MULTIPLY n BY m.`
Division	`r = op1 / op2.`	`DIVIDE m BY n.`
Ganzzahldivision	`r = op1 DIV op2.`	
Modulo	`r = op1 MOD op2.`	
Potenzieren	`r = op1 ** op2.`	

Übliche Regeln: "Punkt vor Strich", "links nach rechts", Klammerung .

mathematische Funktionen

Funktion	Bedeutung
abs , sign	Absolutwert , Vorzeichen (Rückgabe -1, 0, 1)
ceil , floor	kleinste / größte ganze Zahl >= / <= Argument
trunc , frac	ganzzahliger Teil / Dezimalteil des Arguments
sin, cos, tan	trigonometrische Funktionen
exp , log	Exponentialfunktion zur Basis e, nat. Logarithmus
sqrt	Quadratwurzel

Beispiel: `r = sqrt( '5.56' ) .`

Nach "(" und vor ")" muss eine Leerstelle sein!

kaufmännisches Rechnen

- Rechenoperationen mit exakten Ergebnissen
- Felder mit dem **Datentyp p** notwendig
- Berücksichtigung der Dezimalstellen einer gepackten Zahl nur bei eingeschalteter Festpunktarithmetik (siehe Programmeigenschaften)

```
DATA pack TYPE p DECIMALS 0.
pack = 1 / 3 * 3 .
```

Ergebnis ?

Datums- und Zeitberechnungen

```
REPORT z_datum_zeit .

DATA: t_start TYPE t VALUE '080000',
      d_start TYPE d , days TYPE i,
      hours TYPE i .

d_start = sy-datum .
d_start+4(4) = '0101'.
days = sy-datum - d_start + 1.
hours = ( sy-uzeit - t_start ) / 3600.

WRITE: / days, 'Tage',
       'im Jahr', sy-datum(4) ,
       hours, 'Stunden seit 8 Uhr'.
```

Bedingte Verzweigung

```
IF bedingung_1 .
   < anweisungsblock_1 >
[ELSEIF bedingung_2 .
  < anweisungsblock_2 >]...

[ELSE .
  < anweisungsblock_n >]
ENDIF .
```

Mehrfachverzweigung

```
CASE variable .
   WHEN wert_1 [ OR .... ].
    < anweisungsblock_1 >
[  WHEN wert_2 [ OR .... ].  ]...
    < anweisungsblock_2 >

[  WHEN OTHERS .             ]
    < anweisungsblock_n >    ]
ENDCASE .
```

Schleifen

```
WHILE bedingung .
    < anweisungsblock >
ENDWHILE .

DO [ n TIMES ].
    < anweisungsblock >
ENDDO .
```

Bei jedem Schleifendurchlauf wird **sy-index** hoch gezählt.

Iterationsmodell while-Schleife

```
<Erster Fall außerhalb der Schleife
behandeln, i.d.R. wird damit die
Schleifenbedingung initialisieren.>;

while (<Bedingung>) {

  <Bearbeiten des ersten / aktuellen
     Falls.>;
  <Nächster Fall behandeln,
    "Update" auf Schleifenbedingung.>;
}
```

Noch Fragen?