4. Zur Struktur der PS PASCAL

4.1. Einleitende Bemerkungen

- 1953/54 (USA) PS FORTRAN (FORmula TRANslating system)
- 1958/60 (Europa) ALGOL (ALGOrithmic Language)
- Anfang 60er (USA) COBOL (Commercial Business Lang.)
- Mitte 60er von IBM: PL/1
- 70/71 von N. Wirth: PASCAL
- Anfang 70er Ausschreibung der USA (Pentagon): ADA
  Wirth: MODULA; später zu MODULA II verbessert
  ⇨ Turbo–Pascal
- Anfang 70er in Bell–Laboratories: C
- objektorient. Programmierung:
  C++; höhere Versionen von Turbo–P., ab 5.5
  N.Wirth ➔ OBERON
  (siehe Informatik Spektrum, Band 17, Heft 1, 1994, S. 5–10)

4.2. Lexikale Sprachelemente (Vokabular)

lexikales Sprachelement := ZR in der jeweiligen Sprache mit definiertem syntaktischem Aufbau und festgelegter Bedeutung bei ihrer Verwendung;

(1) Bezeichner: (= Namen) ∃ klares Regelwerk;

(2) Grundsymbole: (= Schlüssel***wörter***)

(3) Zahlen:

(4) Zeichenkette:

(5) Operatoren, Begrenzer, technische Zeichen:

Operatoren: + – * / AND OR < > >= usw.
Begrenzer: . , ;
techn. Zeichen: [ ] ( ) usw.

(6) Kommentare:

Programmaufbau für PASCAL einfach:

.

4.3. PASCAL–Elemente für einfache Programme

(1) Anweisungen:

***a) einfache Anweisungen*:**

- **Ergibtanweisung**
- **Prozeduraufrufe** (nicht in jeder PS)

b) strukturierte Anweisungen:

(i) **Verbundanweisung:**

```
BEGIN
    anweisung1;
    anweisung2;
    …;
    anweisungn
END;
```

(ii) **Alternative:** IF–THEN–Anweisung;
CASE–Anweisung; (bzw. Fall–Anweisung)

```
IF bedingung
   THEN anw1
   ELSE  anw2;
```

```
CASE fallausdruck OF
   fallkonstante1: anw1;
   fallkonstante2: anw2;
   …
   fallkonstanten: anwn
   ELSE anw
END;
```

(iii) **Schleifen:**

Zählschleife

FOR zaehlvariable := awert TO ewert DO anweisung;
{DOWNTO}

WHILE–Schleife (Abweisschleife)

WHILE bedingung DO anweisung;

REPEAT–Schleife (Nichtabweisschleife)

REPEAT anweisung UNTIL bedingung;

(2) Ausdrücke:

a) numerische Ausdrücke

b) logische Ausdrücke

Aus ***Operanden*** und ***Operatoren***.

4 Klassen von Operatoren:

arithmetische, logische, Vergleichs-, Mengen–Operatoren

(3) Datenobjekte:

- ***einfache*** und ***strukturierte*** DO

DO = (N, T, W)

- ***Konstanten***
- ***Variablen***

Variablenkonzept

- ***Deklarationszwang*:** Alle DO müssen vereinbart werden!

***a) einfache DO*:**

numerische und nichtnumerische;

INTEGER; REAL CHAR; BOOLEAN

ganzzahlige (numer.) Datentypen:

BYTE	1 Byte	0 … 255
SHORTINT	1 Byte	–128 … 127
INTEGER	2 Byte	–32 768 … 32 767
WORD	2 Byte	0 … 65 535
LONGINT	4 Byte	-2^{31} … $2^{31} - 1$

MAXINT = 32 767; MAXLONGINT = 2 147 483 647

reelle Zahlen:

REAL	6 Byte	$2.9 \cdot 10^{-39}$	bis $1.7 \cdot 10^{38}$
in Abh. von Konfiguration noch:			
SINGLE	4 Byte	$1.5 \cdot 10^{-45}$	bis $3.4 \cdot 10^{38}$
DOUBLE	8 Byte	$5.0 \cdot 10^{-324}$	bis $1.7 \cdot 10^{308}$
EXTENDED	10 Byte	$3.4 \cdot 10^{-4951}$	bis $1.1 \cdot 10^{4932}$

CHAR: 1 Byte 256 Zeichen

Besonderheit von PASCAL:

1. Anwender kann eigene DO definieren.
2. Anwender kann auch eigene Typen definieren.

⇨ **Typvereinbarung:**

```
TYPE   typname = typ;
```

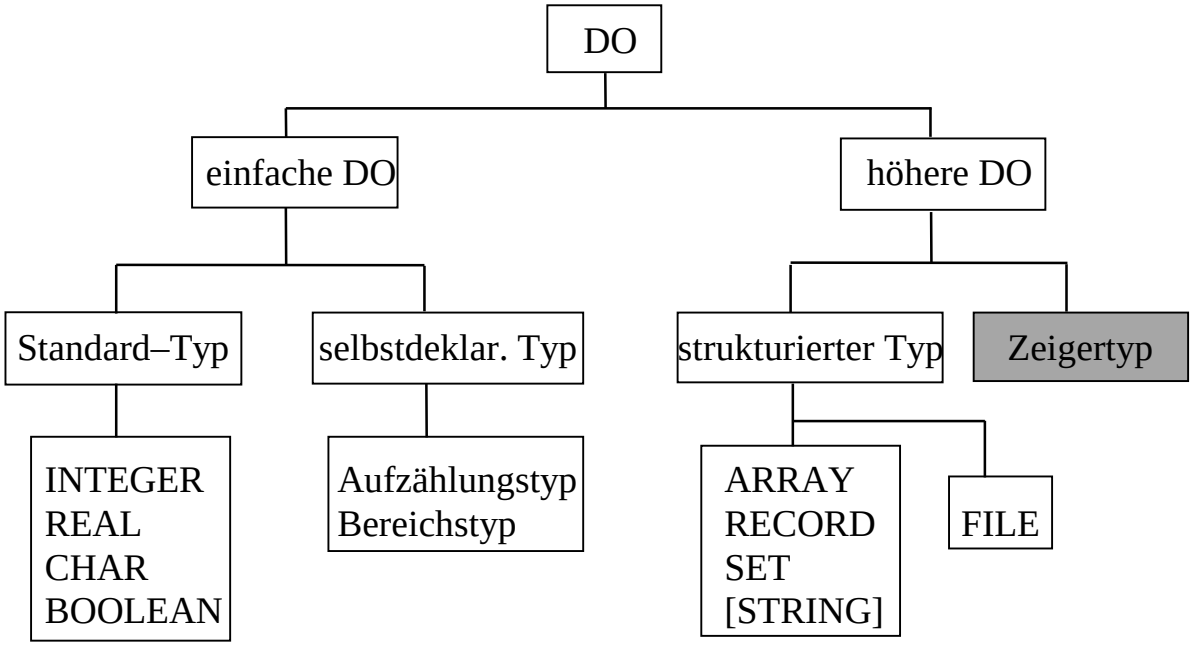
➔ 2 spezielle Typen:

a) ***Aufzählungstyp***
b) ***Bereichstyp*** (Unterbereichstyp)

***b) strukturierte DO*:**

	ARRAY	(Feld)
	RECORD	(Verbund)
	SET	(Menge)
	FILE	(Datei)
{ in Turbo–P.:	STRING	(Zeichenkette) }

***Pointer (Zeiger)*:** dynamisches DO



(4) Vereinbarungen: Im Vereinbarungsteil ➔

1. CONST konstname = konstwert;
2. TYPE typname = typ; {einfacher oder strukturierter Typ}
3. VAR varname : vartyp;
4. Teilalgorithmen (Prozeduren; Funktionen)

4.4. Bemerkungen zum Zahlenrechnen

a) Festpunktzahlen: i.allg. Integer–Objekte

$$z = \pm \sum_{i=-k}^{m} a_i \cdot B^i$$

$$MAXINT := (B-1) \sum_{i=0}^{m} B^i = B^{m+1} - 1$$

$$MININT := -(B^{m+1} - 1)$$

! Im Integer–Bereich gilt Assoziativgesetz nicht!!!

➔ richtig klammern

Bsp.:

```
PROGRAM IntegerTest;

VAR a, b, i : INTEGER;

BEGIN
  a := 500;
  b := 900;
  i  := 100 * b DIV a;
  WRITELN(i);
  READLN
END.
```

b) Echte Brüche: m = –1

B = 2 ➔ Darstellungsintervall $-1 + 2^{-k} \leq z \leq 1 - 2^{-k}$

⇨ reelle Zahl z hieraus durch endlichen Bruch $\underline{z}$ approximiert mit

$$|z - \underline{z}| \leq \tfrac{1}{2} \cdot 2^{-k} = 1/2^{k+1}$$

c) Gleitpunktzahlen: (GPZ)

$$z = \pm m \cdot B^p$$

Mantisse: $|m| = \sum_{i=-1}^{-k} a_i \cdot B^i$

Exponent: $p \in [-E, E]$ ist Integerzahl

⇨ $z = \pm a_{-1}\, a_{-2} \ldots a_{-k} \pm b_{l-1}\, b_{l-2} \ldots b_1\, b_0$
(Mantisse) (Exponent)

bei l Exponentenstellen ➔ $E = B^l - 1$ ist max. Exponent

$Z = B^E \cdot (1 - B^{-k})$ ist größte pos. GPZ

$\underline{Z} = B^{-E} \cdot (1 - B^{-k})$ ist kleinste pos. GPZ

Zahl der Exp.-stellen beeinflußt Zahlenbereich
Zahl der Mantissenstellen – Genauigkeit der Darstellung

Darstellungsintervall ist $-2^E \cdot (1 - 2^{-k}) \leq z \leq 2^E \cdot (1 - 2^{-k})$

$|z| < 2^{-E-k}$ ➔ Darstellung durch 0 (GP–Null)

$|z| \geq 2^{-E-k}$ ➔ Näherung: korrekt gerundete GP–Zahl z'

Approximationsfehler $|z - z'| \leq |z| \cdot 2^{-k}$
ein relativer Fehler mit $|z - z'| / |z| \leq 2^{-k}$

➔ ***Konsequenzen für Arithmetik***:

1. Add. von GPZ unterschiedlicher Größenordnung vermeiden.
2. Subtraktion von benachbarten GPZ vermeiden.
3. Vergleich zweier GPZ kann zu fehlerhaften Entsch. führen!

! Numerischen Probleme ernst nehmen!!!