4.7. Dynamische Variable – das Zeigerkonzept

4.7.1. Vorbemerkungen

■ noch praktisch bedeutsame Lücke:
z.B. alphabet. geordnete Adress-Kartei
als **Feld von Records** → vorher Länge festlegen
als **File von Records** → nur am Ende Eintrag

■ außerdem bisher bei Vereinbarungen
(a) Name und
(b) Typ
➔ mit Def. (beim Compilieren) ein **typspezifischer Sp.-platz** ***über ges. Dauer*** zugewiesen

➔ **statische Variable**

■ Ausweg: **dynamische Variable**

- erst angelegt, wenn sie benötigt werden
- nicht explizit vereinbart, d.h., statt eines Namens erhalten sie einen **Zeiger** : verweist auf eine HS–Adresse, in welcher der Wert der referierten Var. aufbewahrt ist
- Begriff des Bezuges (Referenz, Verweis);
Übertrag in PASCAL-Umgebung:

→ namenlose Schublade ⇔ namenlose, anonyme Var., die irgendwo im adressierb. Sp.-bereich
→ Finger ⇔ Zeiger auf **ADRESSE** dieser Var.
Zeiger kann **auf** ∀ **Sp.-adresse** zeigen – **Var.-name** zeigt (konstant) auf **nur eine Sp.-adresse**

4.7.2. Zeiger und dynamische Variable

■ mit VAR i: Integer; wird Var. vom Typ Integer vereinbart;
in TP auch folg. Deklaration möglich:

```
VAR p : ^Integer;
```

➔ Zeigervariable mit Namen p vereinbart, die vom Typ *„zeigt auf Variable des Typs Integer“* ist – **Zeigertyp**;

physisch: zeigt auf Startadresse eines Speicherplatzes, in dem eine Integer–Variable gespeichert werden kann;

! p selbst ist **nicht** vom Typ Integer !!!

■ allg. Deklarationsvorschrift:

```
VAR zeigervariablenname : ^datentypname;
```

■ auch über Zeiger–Typ–Deklaration:

```
TYPE zeiger = ^Integer;
VAR  p      :  zeiger;
```

➔ Typ–Vereinbarung:

```
TYPE zeigername = ^typname;
```

dadurch wie üblich festgelegt:

a) Menge der Operationen: „Wert“–Zuweisung; Test auf Gleichheit
b) Menge der Werte: alle Speicheradressen, die von Objekten des Typs typname eingenommen werden können; dazu NIL

■ Erzeugen dyn. Objekte:

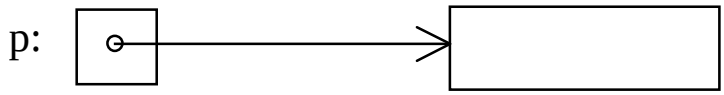
in TP ∃ vordef. Prozedur **NEW(zeigervariablenname);**

Wirkung (für VAR p:^Integer;):

1. NEW(p) weist einer Var. vom Typ Integer entspr. (einen) Speicherplatz auf dem Heap zu.

2. Wert von p zeigt auf diese Var. bzw. dem Zeiger p wird die Adresse des entspr. Speicherplatzes zugewiesen.

NEW(p):



p^ : Objekt vom Typ Integer

■ Frage: Wie auf Var.(Objekt) zugreifen, auf die (das) gezeigt wird?
im Bsp. mit p^, allgemein mit

zeigervariablenname^

- dadurch **implizit** Bezeichnung der Var. eingeführt, auf die der Wert von zeigervariablenname verweist

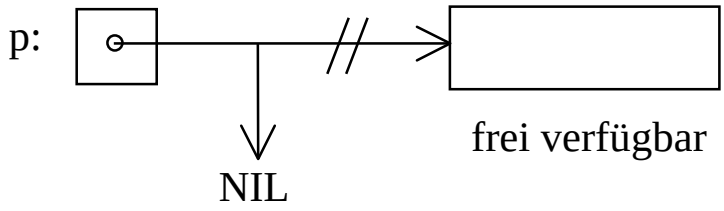
- Variable p^ bzw. zeigervariablenname^ **nicht deklariert**

- **Sprechweise:**

  Der mit NEW(p) reservierte Sp.-bereich wird ebenso wie der Bezeichner p^ als **dynamische Variable** bezeichnet.

■ Löschen dynamischer Objekte (*Dereferenzierung)*:

DISPOSE(p): {als Gegenstück zu NEW}



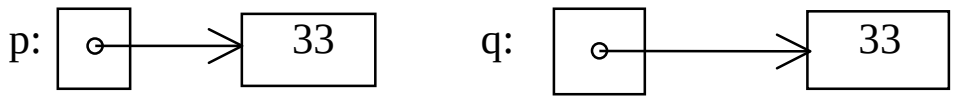
■ zur Arbeit mit Zeigern und Zeigervariablen
sei VAR p, q: ^Integer;
seien die zu p und q zugehörigen dyn. Variablen p^ und q^ mit NEW(p) bzw. NEW(q) angelegt;

- nun können p^ und q^ auch Dateninhalte zugewiesen werden:

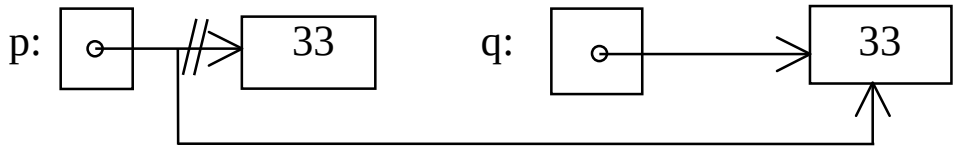
p^:= 20; q^:= 33;

danach auch **p^ := q^ ;** ➔ folg. Situation:

Zeiger p zeigt auf Var. p^, deren Dateninhalt 33 ist;
Zeiger q zeigt auf Var. q^, deren Dateninhalt 33 ist;



ABER: **p := q;** ➔ Zeiger p zeigt auf gleiche Adresse wie Zeiger q !!!



Bem.: Bei Wertzuweisung unterscheiden zw.
Zuweisung an Zeigervariable und
Zuweisung an dynamische Variable (Bezugsvar.).

■ Zeigervar. auch bei benutzerdef. Datentypen:

z.B., wenn ein Zeiger auf eine Var. vom Typ Person zeigen soll
➔

```
TYPE zeiger  = ^person;
       person = RECORD
                   name: String[30];
                   alter : Byte
                END;

VAR  p : zeiger;
```

Problem:

In 1. Zeile Objekt angesprochen, das noch nicht deklariert !

➔ in TP : Zeiger – einziger Datentyp, der **rekursiv** def. werden darf bzw. **vor** der Def. des Objektes, auf das er zeigen soll.

wenn nun NEW(p) ➔
eine Var. p^ mit 2 Elementen angelegt:

p^.name und p^.alter

➔ Zeiger p zeigt auf die Variable p^ :