universitätbonn

Verbunde - structs

Verbunde (records, structs)

- Arrays modellieren also Beziehungen zwischen Elementen gleichen Typs
- Oft bestehen aber auch Beziehungen zwischen Werten unterschiedlichen Typs
  - Etwa zwischen Name und Monatsverdienst eines Beschäftigten
- Wir verbinden zusammengehörige Daten unterschiedlichen Typs zu einem **Verbund** (record, structure, struct)
- **Beispiel:** Stammdaten

Name	"Mustermann"
Vorname	"Martin"
GebTag	10
GebMonat	05
GebJahr	1930
Familienstand	"verheiratet"
...	...

Verbunde (records, structs)

- Übliche Syntax zur Auswahl: Punkt-Notation
  - Beispiel:

Sei ein konkretes Stammdatenblatt `s` gegeben. Dann ist

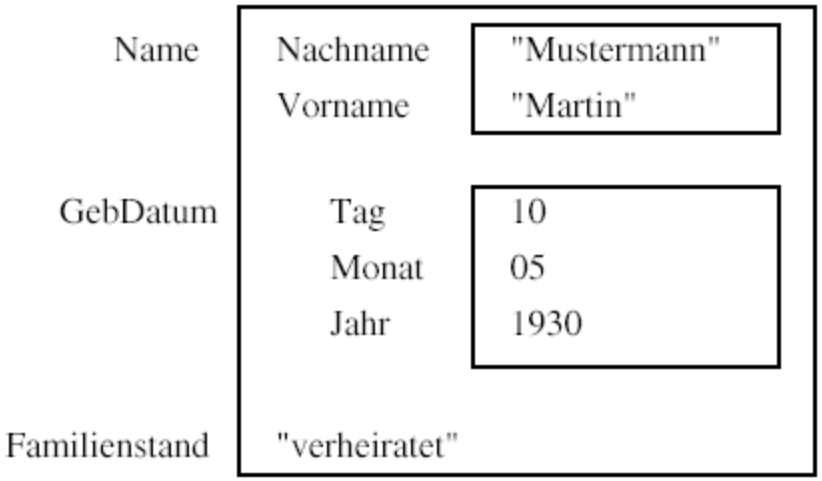
```
s.Name = "Mustermann"
```

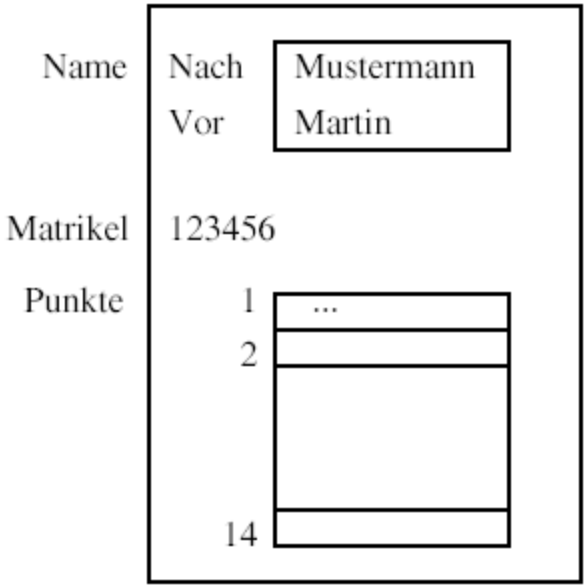
der Wert der Komponente `Name` von `s`. Entsprechend gilt `s.GebTag = 10` usw.

- Komponenten eines Verbunds können von beliebigem Typ sein
  - Also auch wieder Verbunde, Reihungen, etc.

Verbunde (records, structs)

- **Beispiele komplexer Verbunde:**





Structs in C

- Die Deklaration von Verbunden in C ist sehr simpel. Sie erfolgt über das Schlüsselwort *struct*:
  - struct StrukturName;
- Beispiele

```
struct StrukturName {
  int ErstesAttribut;
  char *ZweitesAttribut;
  float DrittesAttribut; };
```

```
/* Definition der Struktur: */
struct StrukturName { int i; char c; };
/* Deklaration der Struktur-Variable: */
struct StrukturName StrukturVariablenName;
/* Initialisierung der Struktur-Variable: */
StrukturVariablenName.i = 1;
StrukturVariablenName.c = 'x';
```

Verwendung von typedef

- Mit der *Anweisung* *typedef* lässt sich in den C-basierten Programmiersprachen die Übersichtlichkeit des Quellcodes verbessern, da es dadurch möglich wird, bei der Deklaration von Struktur-Variablen auf das Schlüsselwort *struct* zu verzichten
- Beispiel

```
/* Definition einer Struktur: */
 struct StrukturName { int i; int j; };
/* Typ-Definition mit typedef: */
typedef StrukturName TypName;
/* Übersichtliche Deklaration einer Struktur-Variable: */
TypName StrukturVariablenName;
```

Zeigervariablen auf structs

```
struct point { int x; int y; } my_point;
struct point *p = &my_point; /* To declare p as a pointer of type struct point */
(*p).x = 8; /* To access the first member of the struct */
p->x = 8; /* Another way to access the first member of the struct */
```

Rekursion auf Datentypebene

- Structs, die Zeigervariablen auf das struct enthalten, von dem sie ein Teil sind
  - Diese zunächst seltsam anmutende Idee ist zentral für viele komplexe Datenstrukturen
  - Verkettete Listen, Bäume, …

Verkettete Listen

- Liste besteht aus Knoten, die Datenfeld beinhalten
  - Anwendungen ähnlich zu arrays

```
typedef struct NODE {
  struct NODE *next;
  int data;
 } NODE;
```

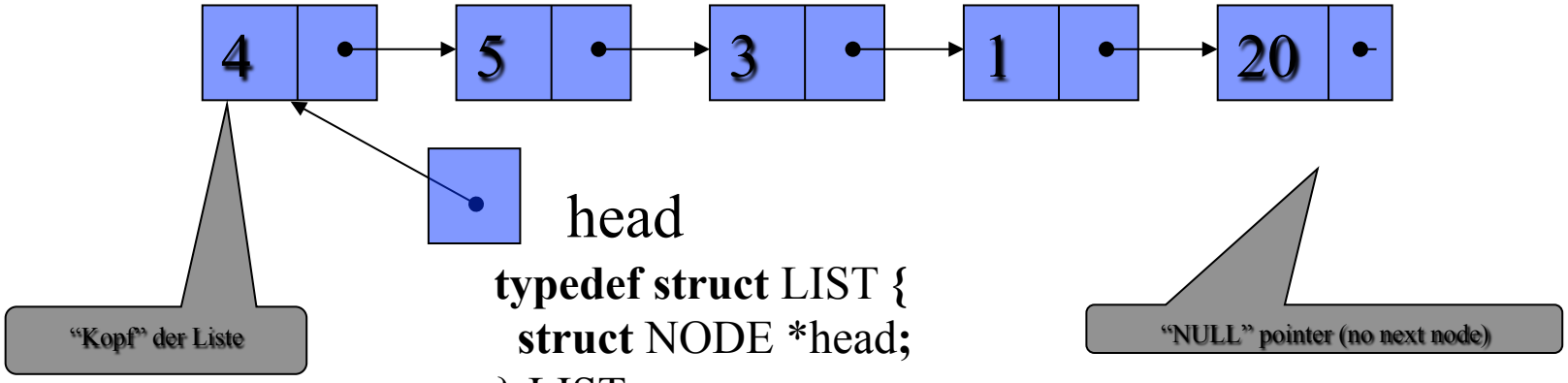
```
typedef struct LIST {
  struct NODE *head;
} LIST;
```

Einfach verkettete Liste

- Einfach verkettete Liste einfachster Fall einer „rekursiven“ Datenstruktur
- Repräsentieren im folgenden Liste durch Zeigervariable auf den Kopf (head) der Liste

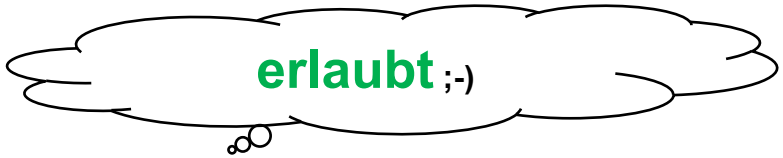
```
typedef struct NODE {
  struct NODE *next;
  int data;
} NODE;
```

Schematisches Beispiel:




```
typedef struct LIST {
  struct NODE *head;
} LIST;
```

Einfach verkettete Liste



- Aufbau erfordert dynamische Speicherverwaltung
  - Siehe später

universitätbonn

Modellierung des Enthaltenseins

Modellierung des Enthaltenseins - Referenzen

- Ein Verbund kann in einem anderen enthalten sein
  - Vgl. Beispiel von Datum und Stammdatenblatt
- Diese Beziehung des Enthaltenseins (containment) kann auf zweierlei Arten modelliert werden
  - Als Enthaltensein durch Wert (by value)
  - Als Enthaltensein durch Referenz (by reference)

Modellierung des Enthaltenseins - Referenzen

- **Beispiel:**
  - Studentin Musterfrau belegt zwei verschiedene Übungen
    - ð Übung durch Verbund realisiert
      - Enthält u.a. String Übungsleiter, Tabelle (Reihung) mit Stammdaten des Studierenden sowie einer Tabelle mit Punkten
    - ð Stammdatenblatt Musterfrau also in zwei verschiedenen Verbunden realisiert
    - ð Wenn dies durch *Enthaltensein durch Wert* modelliert wird, dann existieren zwei separate Exemplare des Stammdatenblatts Wenn dies durch *Enthaltensein durch Referenz* modelliert wird, dann existiert nur **ein** Exemplar des Stammdatenblatts Musterfrau, auf das beide Verbunden ve