5.1.F Datenstrukturen II/Records – Ergänzungen und Bilder

5.1.F.1 Record enthält Record-Typ

Wie beschrieben kann ein Record-Typ als Komponente ein anderes Record enthalten:

```
1  struct adresstyp{
2     char strasse[30];
3     int hausnummer;
4  };
5  struct persontyp{
6     char vorname[30];
7     char nachname[30];
8     int gebdat;
9     struct adresstyp adresse;
10 };
```

Es gibt aber auch die Möglichkeit, den inneren Record-Typ direkt im äußeren Record-Typ zu vereinbaren:

```
1  struct persontyp{
2     char vorname[30];
3     char nachname[30];
4     int gebdat;
5     struct
6     {
7        char strasse[30];
8        int hausnummer;
9     } adresse;
10 };
```

Das Etikett `adresstyp` fehlt hier in Zeile 5, weil es nicht gebraucht wird. Hier kann man allerdings nicht mehr einzeln eine Variable des inneren Typs anlegen. Der Zugriff ist wie bei der anderen Variante:

```
1  struct persontyp direktor;
2  strcpy(direktor.adresse.strasse,"Schulstrasse");
3  direktor.adresse.hausnummer=20;
```

5.1.F.2 Record enthält anonymen Record-Typ

Seit C11 ist es möglich, dem inneren Record keinen Namen zu geben; der Zugriff auf die Komponenten des inneren Records erfolgt dann mit nur einem Punkt-Operator (anonymes Record):

```
1  struct persontyp{
2     char vorname[30];
3     char nachname[30];
4     int gebdat;
5     struct
6     {
7        char strasse[30];
8        int hausnummer;
9     };
10 };
```

Hier fehlt nicht nur das Etikett `adresstyp` in Zeile 5, sondern auch der Komponenten-Name `adresse` in Zeile 9. Der Zugriff ist jetzt einfacher; man braucht nur noch einen Punkt-Operator statt zwei:

```
1 struct persontyp direktor;
2 strcpy(direktor.strasse,"Schulstrasse");
3 direktor.hausnummer=20;
```