WBS TRAINING

Programmierung(2)

UNSER ♥ SCHLÄGT FÜR BILDUNG

Agenda

- **Strukturen**
  - Definition
  - Motivation
- **Strukturtyp**
  - Deklaration und Definition
- **Struktur-Variable**
  - Deklaration, Definition, Initialisierung und Ausgabe
- **Struktur-Array**
  - Deklaration, Definition, Initialisierung und Ausgabe
- **Strukturen in Strukturen**
  - Strukturtyp A in Strukturtyp B
  - Strukturtyp A in Strukturtyp A (siehe Fachpraktische Anwendungen)
- **Fachpraktische Anwendungen**

Struktur – Definition

- Wir haben bisher „zusammengehörige“ Variablen mittels eines **Arrays** zusammenfassen können. Diese Variablen mussten dann allerdings alle **vom selben Typ** (int, char, float …) sein.
- **Strukturen** erlauben nun das Zusammenfassen mehrerer „zusammengehöriger“ Variablen auch dann, wenn diese von **unterschiedlichem Typ** sind.
- Jede einzelne der zusammengefassten Variablen wird als **Strukturkomponente** bezeichnet.
- Die Zusammenfassung mehrerer Strukturkomponenten kann selbst wieder als ein neuer (vom Programmierer quasi selbsterschaffener) **Typ** betrachtet werden. Diesen nennen wir **Strukturtyp**.
- Variablen eines solchen Strukturtyps, werden dann entsprechend **Strukturvariablen** genannt.
- **Strukturvariablen** erinnern aus Sicht des Datenbank/SQL-Bausteins an **Datensätze**, die ebenfalls mehrere zusammengehörige Informationen unterschiedlichen Typs zusammenfassen.
- Betrachtet man hingegen ein ganzes Array von Strukturvariablen, so erinnert dies an eine Tabelle.
- Ferner gilt: Aus Sicht der „**Objektorientierten Programmierung**“ erinnern Strukturtypen an Klassen und Strukturen an „**Objekte**“. Dies wird dann aber erst im Baustein zu C# thematisiert.

Struktur – Motivation

- Wenn wir (z.B.) Baujahr, PS-zahl, Kilometerstand und Preis ein und des selben Autos abspeichern wollen, dann benötigen wir 4 getrennte Variablen. Im Quellcode wäre dann aber nicht unmittelbar ersichtlich, dass diese Variablen vom selben Auto sprechen => hier helfen **Strukturvariablen**.

- Um Strukturvariablen nutzen zu können, werden wir einen entsprechenden **Strukturtypen** einführen. Dies ermöglicht uns dann aber auch, ganze Arrays von diesem Typ zu deklarieren.

- Ferner kann mittels einzelner **Strukturkomponenten** auf andere Strukturvariablen verwiesen werden. Auf diese Weise können wir Beziehungen zwischen Strukturvariablen darstellen.

Struktur – **Beispielaufgabe (1)**

Aufgabenstellung

- Das Programm führt zunächst einen **Strukturtypen** PERSON ein, für den die **Strukturkomponenten** ID, Vorname, Nachname und Gehalt angelegt werden.
- Anschließend wird eine **Strukturvariable** p vom Typ PERSON deklariert, definiert und initialisiert.
- Der Inhalt der Strukturkomponenten soll anschließend zur Kontrolle auf der Konsole ausgegeben werden. Danach endet das Programm.

Hinweis:

Erneut werden wir auf eine Darstellung der Lösung als PAP, Struktogramm oder Pseudocode **verzichten** können, da Strukturen keine eigene Notation besitzen. Der verwendete Strukturtyp (mit samt der verwendeten Strukturkomponenten) darf aber natürlich dennoch im PAP, Struktogramm oder Pseudocode erwähnt werden. Wie dies geschieht, unterliegt dann aber keiner strengen Regel.

Im Zweifel werden Sie schlicht die Notation der folgenden Quellcodes verwenden können.

(Siehe Aufgabe 05_05_01)

Struktur – **Beispielaufgabe (1)** – **Quellcode**

```
#include<stdio.h>
#include<windows.h>
#include<string.h>

typedef struct
{
    int id;
    char vorname[31];
    char nachname[51];
    float gehalt;
}
PERSON;


int main(void)
{
    system("chcp.com 1252");
    system("cls");

    PERSON p;
    p.id=4711;
    strcpy(p.vorname,"Petra");
    strcpy(p.nachname,"Mustermann");
    p.gehalt=2222.99;

    printf("Kontrollausgabe:\n\nID: %d\nVorname: %s\nNachname: %s\nGehalt: %.2f €",p.id,p.vorname,p.nachname,p.gehalt);

    printf("\n\n\n\n");
    system("pause");
    return 0;
}
```

WBS TRAINING

Struktur – **Beispielaufgabe (1)** – **Quellcode**

```
#include<stdio.h>
#include<windows.h>
#include<string.h>

typedef struct
{
    int id;
    char vorname[31];
    char nachname[51];
    float gehalt;
}
PERSON;


int main(void)
{
    system("chcp.com 1252");
    system("cls");

    PERSON p;
    p.id=4711;
    strcpy(p.vorname,"Petra");
    strcpy(p.nachname,"Mustermann");
    p.gehalt=2222.99;

    printf("Kontrollausgabe:\n\nID: %d\nVorname: %s\nNachname: %s\nGehalt: %.2f €",p.id,p.vorname,p.nachname,p.gehalt);

    printf("\n\n\n\n");
    system("pause");
    return 0;
}
```

Mit **typedef struct** teilen wir mit, dass ein **Strukturtyp** eingeführt wird.

WBS TRAINING

Struktur – **Beispielaufgabe (1)** – **Quellcode**

```
#include<stdio.h>
#include<windows.h>
#include<string.h>

typedef struct
{
    int id;
    char vorname[31];
    char nachname[51];
    float gehalt;
}
PERSON;


int main(void)
{
    system("chcp.com 1252");
    system("cls");

    PERSON p;
    p.id=4711;
    strcpy(p.vorname,"Petra");
    strcpy(p.nachname,"Mustermann");
    p.gehalt=2222.99;

    printf("Kontrollausgabe:\n\nID: %d\nVorname: %s\nNachname: %s\nGehalt: %.2f €",p.id,p.vorname,p.nachname,p.gehalt);

    printf("\n\n\n\n");
    system("pause");
    return 0;
}
```

Hinter der geschwungenen Klammer wird der (selbstgewählte) **Name** des Strukturtyps deklariert.
Achtung: *Semikolon* dahinter nicht vergessen!

WBS TRAINING

Struktur – **Beispielaufgabe (1)** – **Quellcode**

```
#include<stdio.h>
#include<windows.h>
#include<string.h>

typedef struct
{
    int id;
    char vorname[31];
    char nachname[51];
    float gehalt;
}
PERSON;


int main(void)
{
    system("chcp.com 1252");
    system("cls");

    PERSON p;
    p.id=4711;
    strcpy(p.vorname,"Petra");
    strcpy(p.nachname,"Mustermann");
    p.gehalt=2222.99;

    printf("Kontrollausgabe:\n\nID: %d\nVorname: %s\nNachname: %s\nGehalt: %.2f €",p.id,p.vorname,p.nachname,p.gehalt);

    printf("\n\n\n\n");
    system("pause");
    return 0;
}
```

Die 4 **Strukturkomponenten** des Strukturtyps PERSON
Achtung: werden jeweils durch Semikolon getrennt

WBS TRAINING

Struktur – **Beispielaufgabe (1) – Quellcode**

```
#include<stdio.h>
#include<windows.h>
#include<string.h>

typedef struct
{
   int id;
   char vorname[31];
   char nachname[51];
   float gehalt;
}
PERSON;


int main(void)
{
   system("chcp.com 1252");
   system("cls");

   PERSON p;
   p.id=4711;
   strcpy(p.vorname,"Petra");
   strcpy(p.nachname,"Mustermann");
   p.gehalt=2222.99;

   printf("Kontrollausgabe:\n\nID: %d\nVorname: %s\nNachname: %s\nGehalt: %.2f €",p.id,p.vorname,p.nachname,p.gehalt);

   printf("\n\n\n\n");
   system("pause");
   return 0;
}
```

Unterhalb der Präprozessoranweisungen und oberhalb des main => Strukturvariablen werden üblicherweise **global** eingeführt, da dieser Typ auch für Funktionen nutzbar seien soll.
(**Erläuterung:** auch die Typen int, char, float … gelten ja global für den gesamten Quellcode)

WBS TRAINING

Struktur – **Beispielaufgabe (1)** – **Quellcode**

```
#include<stdio.h>
#include<windows.h>
#include<string.h>

typedef struct
{
    int id;
    char vorname[31];
    char nachname[51];
    float gehalt;
}
PERSON;


int main(void)
{
    system("chcp.com 1252");
    system("cls");

    PERSON p;
    p.id=4711;
    strcpy(p.vorname,"Petra");
    strcpy(p.nachname,"Mustermann");
    p.gehalt=2222.99;

    printf("Kontrollausgabe:\n\nID: %d\nVorname: %s\nNachname: %s\nGehalt: %.2f €",p.id,p.vorname,p.nachname,p.gehalt);

    printf("\n\n\n\n");
    system("pause");
    return 0;
}
```

Deklaration und Definition der **Strukturvariable** p vom Typ Person

WBS TRAINING

Struktur – **Beispielaufgabe (1)** – **Quellcode**

```
#include<stdio.h>
#include<windows.h>
#include<string.h>

typedef struct
{
    int id;
    char vorname[31];
    char nachname[51];
    float gehalt;
}
PERSON;


int main(void)
{
    system("chcp.com 1252");
    system("cls");

    PERSON p;
    p.id=4711;
    strcpy(p.vorname,"Petra");
    strcpy(p.nachname,"Mustermann");
    p.gehalt=2222.99;

    printf("Kontrollausgabe:\n\nID: %d\nVorname: %s\nNachname: %s\nGehalt: %.2f €",p.id,p.vorname,p.nachname,p.gehalt);

    printf("\n\n\n\n");
    system("pause");
    return 0;
}
```

Initialisierung der **Strukturkomponente** id (innerhalb der Strukturvariable p)
p teilt mit, welche **Strukturvariable** gemeint ist
id teilt mit, welche **Strukturkomponente** angesprochen werden soll
mittels „**Punktoperator**“(**.**) kann p – anschaulich gesprochen - „betreten“ werden

WBS TRAINING

Struktur – **Beispielaufgabe (1)** – **Quellcode**

```
#include<stdio.h>
#include<windows.h>
#include<string.h>

typedef struct
{
    int id;
    char vorname[31];
    char nachname[51];
    float gehalt;
}
PERSON;


int main(void)
{
    system("chcp.com 1252");
    system("cls");

    PERSON p;
    p.id=4711;
    strcpy(p.vorname,"Petra");
    strcpy(p.nachname,"Mustermann");
    p.gehalt=2222.99;

    printf("Kontrollausgabe:\n\nID: %d\nVorname: %s\nNachname: %s\nGehalt: %.2f €",p.id,p.vorname,p.nachname,p.gehalt);

    printf("\n\n\n\n");
    system("pause");
    return 0;
}
```

Initialisierung der **Strukturkomponenten** vor- und nachname.
Da es sich um Strings handelt, muss natürlich die Stringfunktion **strcpy** verwendet werden

WBS TRAINING

Struktur – **Beispielaufgabe (1)** – **Quellcode**

```
#include<stdio.h>
#include<windows.h>
#include<string.h>

typedef struct
{
    int id;
    char vorname[31];
    char nachname[51];
    float gehalt;
}
PERSON;


int main(void)
{
    system("chcp.com 1252");
    system("cls");

    PERSON p;
    p.id=4711;
    strcpy(p.vorname,"Petra");
    strcpy(p.nachname,"Mustermann");
    p.gehalt=2222.99;

    printf("Kontrollausgabe:\n\nID: %d\nVorname: %s\nNachname: %s\nGehalt: %.2f €",p.id,p.vorname,p.nachname,p.gehalt);

    printf("\n\n\n\n");
    system("pause");
    return 0;
}
```

Initialisierung der **Strukturkomponente** gehalt.

WBS TRAINING

Struktur – **Beispielaufgabe (1)** – **Quellcode**

```
#include<stdio.h>
#include<windows.h>
#include<string.h>

typedef struct
{
    int id;
    char vorname[31];
    char nachname[51];
    float gehalt;
}
PERSON;


int main(void)
{
    system("chcp.com 1252");
    system("cls");

    PERSON p;
    p.id=4711;
    strcpy(p.vorname,"Petra");
    strcpy(p.nachname,"Mustermann");
    p.gehalt=2222.99;

    printf("Kontrollausgabe:\n\nID: %d\nVorname: %s\nNachname: %s\nGehalt: %.2f €",p.id,p.vorname,p.nachname,p.gehalt);

    printf("\n\n\n\n");
    system("pause");
    return 0;
}
```

Ausgabe aller Strukturkomponenten

WBS TRAINING

Struktur – **Beispielaufgabe (1)** – **Quellcode**

```
#include<stdio.h>
#include<windows.h>
#include<string.h>

typedef struct
{
    int id;
    char vorname[31];
    char nachname[51];
    float gehalt;
}
PERSON;


int main(void)
{
    system("chcp.com 1252");
    system("cls");

    PERSON p;
    p.id=4711;
    strcpy(p.vorname,"Petra");
    strcpy(p.nachname,"Mustermann");
    p.gehalt=2222.99;

    printf("Kontrollausgabe:\n\nID: %d\nVorname: %s\nNachname: %s\nGehalt: %.2f €",p.id,p.vorname,p.nachname,p.gehalt);

    printf("\n\n\n\n");
    system("pause");
    return 0;
}
```

```
Kontrollausgabe:

ID: 4711
Vorname: Petra
Nachname: Mustermann
Gehalt: 2222.99 €
```

WBS TRAINING

Struktur – **Beispielaufgabe (2)**

Aufgabenstellung

- Das Programm führt zunächst einen **Strukturtypen** AUTO ein, für den die **Strukturkomponenten** kennzeichen und baujahr angelegt werden.
- Anschließend wird ein 3-elementiges **Array** vom Typ AUTO deklariert und definiert.
- In einer Eingabeschleife wird das Array per User-Abfrage **initialisiert**.
- Nach der Schleife startet zur Kontrolle eine **Ausgabeschleife**, die für alle 3 Elemente jeweils die beiden Strukturkomponentenwerte auf der Konsole ausgibt.
- Anschließend endet das Programm.

Struktur – **Beispielaufgabe (2)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    char kennzeichen[31];
    int baujahr;
}
AUTO;


int main(void)
{
    int i;
    AUTO autoArray[3];

    for(i=0;i<3;i++)
    {
        printf("Geben Sie bitte das Kennzeichen ein: ");
        fflush(stdin);
        scanf("%[^\n]",autoArray[i].kennzeichen);

        printf("Geben Sie bitte das Baujahr ein: ");
        fflush(stdin);
        scanf("%d",&autoArray[i].baujahr);
        printf("\n");
    }

    for(i=0;i<3;i++) printf("Kennzeichen: %s\nBaujahr: %d\n\n",autoArray[i].kennzeichen,autoArray[i].baujahr);
    return 0;
}
```

Struktur – **Beispielaufgabe (2)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
   char kennzeichen[31];
   int baujahr;
}
AUTO;


int main(void)
{
   int i;
   AUTO autoArray[3];

   for(i=0;i<3;i++)
   {
      printf("Geben Sie bitte das Kennzeichen ein: ");
      fflush(stdin);
      scanf("%[^\n]",autoArray[i].kennzeichen);

      printf("Geben Sie bitte das Baujahr ein: ");
      fflush(stdin);
      scanf("%d",&autoArray[i].baujahr);
      printf("\n");
   }

   for(i=0;i<3;i++) printf("Kennzeichen: %s\nBaujahr: %d\n\n",autoArray[i].kennzeichen,autoArray[i].baujahr);
   return 0;
}
```

Deklaration und Definition des **Strukturtyps** AUTO

WBS TRAINING

Struktur – **Beispielaufgabe (2)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
   char kennzeichen[31];
   int baujahr;
}
AUTO;


int main(void)
{
   int i;
   AUTO autoArray[3];

   for(i=0;i<3;i++)
   {
      printf("Geben Sie bitte das Kennzeichen ein: ");
      fflush(stdin);
      scanf("%[^\n]",autoArray[i].kennzeichen);

      printf("Geben Sie bitte das Baujahr ein: ");
      fflush(stdin);
      scanf("%d",&autoArray[i].baujahr);
      printf("\n");
   }

   for(i=0;i<3;i++) printf("Kennzeichen: %s\nBaujahr: %d\n\n",autoArray[i].kennzeichen,autoArray[i].baujahr);
   return 0;
}
```

Deklaration und Definition eines **Arrays** vom Typ AUTO

Struktur – **Beispielaufgabe (2)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    char kennzeichen[31];
    int baujahr;
}
AUTO;


int main(void)
{
    int i;
    AUTO autoArray[3];

    for(i=0;i<3;i++)
    {
        printf("Geben Sie bitte das Kennzeichen ein: ");
        fflush(stdin);
        scanf("%[^\n]",autoArray[i].kennzeichen);

        printf("Geben Sie bitte das Baujahr ein: ");
        fflush(stdin);
        scanf("%d",&autoArray[i].baujahr);
        printf("\n");
    }

    for(i=0;i<3;i++) printf("Kennzeichen: %s\nBaujahr: %d\n\n",autoArray[i].kennzeichen,autoArray[i].baujahr);
    return 0;
}
```

Initialisierung der **Strukturkomponenten** per User-Abfrage

WBS TRAINING

Struktur – **Beispielaufgabe (2)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    char kennzeichen[31];
    int baujahr;
}
AUTO;


int main(void)
{
    int i;
    AUTO autoArray[3];

    for(i=0;i<3;i++)
    {
        printf("Geben Sie bitte das Kennzeichen ein: ");
        fflush(stdin);
        scanf("%[^\n]",autoArray[i].kennzeichen);

        printf("Geben Sie bitte das Baujahr ein: ");
        fflush(stdin);
        scanf("%d",&autoArray[i].baujahr);
        printf("\n");
    }

    for(i=0;i<3;i++) printf("Kennzeichen: %s\nBaujahr: %d\n\n",autoArray[i].kennzeichen,autoArray[i].baujahr);
    return 0;
}
```

Ausgabe der **Strukturkomponentenwerte**

Struktur – **Beispielaufgabe (2)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
   char kennzeichen[31];
   int baujahr;
}
AUTO;


int main(void)
{
   int i;
   AUTO autoArray[3];

   for(i=0;i<3;i++)
   {
      printf("Geben Sie bitte das Kennzeichen ein: ");
      fflush(stdin);
      scanf("%[^\n]",autoArray[i].kennzeichen);

      printf("Geben Sie bitte das Baujahr ein: ");
      fflush(stdin);
      scanf("%d",&autoArray[i].baujahr);
      printf("\n");
   }

   for(i=0;i<3;i++) printf("Kennzeichen: %s\nBaujahr: %d\n\n",autoArray[i].kennzeichen,autoArray[i].baujahr);
   return 0;
}
```

```
Geben Sie bitte das Kennzeichen ein: W-TE 21
Geben Sie bitte das Baujahr ein: 1966

Geben Sie bitte das Kennzeichen ein: W-MN 274
Geben Sie bitte das Baujahr ein: 1980

Geben Sie bitte das Kennzeichen ein: B-AP 1
Geben Sie bitte das Baujahr ein: 1999

Kennzeichen: W-TE 21
Baujahr: 1966

Kennzeichen: W-MN 274
Baujahr: 1980

Kennzeichen: B-AP 1
Baujahr: 1999
```

WBS TRAINING

Struktur – **Beispielaufgabe (3)**

Aufgabenstellung

- Das Programm führt zunächst die beiden folgenden **Strukturtypen** ein:
  - PERSON (id und nachname) sowie
  - AUTO (kennzeichen und besitzer), wobei **besitzer** seinerseits auf eine Strukturvariable vom Typ PERSON verweist
- Anschließend wird eine Variable **p** vom Typ PERSON initialisiert.
- Daraufhin wird eine Variable **a** vom Typ AUTO initialisiert, wobei **a.besitzer** auf **p** verweist.
- Es folgt eine erste **Kontrollausgabe** von Kennzeichen und Besitzer-Nachnamen des Autos a.
- Im Anschluss wird der Nachname von p **überschrieben**.
- Es folgt eine zweite **Kontrollausgabe**, bei der erneut Kennzeichen und Besitzer-Nachnamen des Autos a ausgegeben werden. Überprüft werden soll, ob die Änderung des Nachnamens von p auch die gewünschten Konsequenzen für die Strukturkomponente besitzer hatte. (besitzer verweist ja auf p, also müsste sich entsprechend auch der Nachname des Besitzers geändert haben.)

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    int id;
    char nachname[31];
}
PERSON;

typedef struct
{
    char kennzeichen[31];
    PERSON* besitzer;
}
AUTO;

int main(void)
{
    PERSON p;
    p.id=1234;
    strcpy(p.nachname,"Mustermann");

    AUTO a;
    strcpy(a.kennzeichen,"A-BC 999");
    a.besitzer=&p;

    printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    strcpy(p.nachname,"Schulze");
    printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    return 0;
}
```

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
   int id;
   char nachname[31];
}
PERSON;

typedef struct
{
   char kennzeichen[31];
   PERSON* besitzer;
}
AUTO;

int main(void)
{
   PERSON p;
   p.id=1234;
   strcpy(p.nachname,"Mustermann");

   AUTO a;
   strcpy(a.kennzeichen,"A-BC 999");
   a.besitzer=&p;

   printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
   strcpy(p.nachname,"Schulze");
   printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
   return 0;
}
```

Deklaration und Definition des **Strukturtyps** PERSON

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    int id;
    char nachname[31];
}
PERSON;

typedef struct
{
    char kennzeichen[31];
    PERSON* besitzer;
}
AUTO;

int main(void)
{
    PERSON p;
    p.id=1234;
    strcpy(p.nachname,"Mustermann");

    AUTO a;
    strcpy(a.kennzeichen,"A-BC 999");
    a.besitzer=&p;

    printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    strcpy(p.nachname,"Schulze");
    printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    return 0;
}
```

Deklaration und Definition des **Strukturtyps** AUTO

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
   int id;
   char nachname[31];
}
PERSON;

typedef struct
{
   char kennzeichen[31];
   PERSON* besitzer;
}
AUTO;

int main(void)
{
   PERSON p;
   p.id=1234;
   strcpy(p.nachname,"Mustermann");

   AUTO a;
   strcpy(a.kennzeichen,"A-BC 999");
   a.besitzer=&p;

   printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
   strcpy(p.nachname,"Schulze");
   printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
   return 0;
}
```

Deklaration und Definition der Strukturkomponente besitzer vom Typ **PERSON-Pointer**

WBS TRAINING

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    int id;
    char nachname[31];
}
PERSON;

typedef struct
{
    char kennzeichen[31];
    PERSON* besitzer;
}
AUTO;

int main(void)
{
    PERSON p;
    p.id=1234;
    strcpy(p.nachname,"Mustermann");

    AUTO a;
    strcpy(a.kennzeichen,"A-BC 999");
    a.besitzer=&p;

    printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    strcpy(p.nachname,"Schulze");
    printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    return 0;
}
```

Deklaration und Definition der **Strukturvariable** p vom Typ PERSON

WBS TRAINING

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    int id;
    char nachname[31];
}
PERSON;

typedef struct
{
    char kennzeichen[31];
    PERSON* besitzer;
}
AUTO;

int main(void)
{
    PERSON p;
    p.id=1234;
    strcpy(p.nachname,"Mustermann");

    AUTO a;
    strcpy(a.kennzeichen,"A-BC 999");
    a.besitzer=&p;

    printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    strcpy(p.nachname,"Schulze");
    printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    return 0;
}
```

Initialisierung der **Strukturkomponenten** von p

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    int id;
    char nachname[31];
}
PERSON;

typedef struct
{
    char kennzeichen[31];
    PERSON* besitzer;
}
AUTO;

int main(void)
{
    PERSON p;
    p.id=1234;
    strcpy(p.nachname,"Mustermann");

    AUTO a;
    strcpy(a.kennzeichen,"A-BC 999");
    a.besitzer=&p;

    printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    strcpy(p.nachname,"Schulze");
    printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    return 0;
}
```

Deklaration und Definition der **Strukturvariable** a vom Typ AUTO

WBS TRAINING

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    int id;
    char nachname[31];
}
PERSON;

typedef struct
{
    char kennzeichen[31];
    PERSON* besitzer;
}
AUTO;

int main(void)
{
    PERSON p;
    p.id=1234;
    strcpy(p.nachname,"Mustermann");

    AUTO a;
    strcpy(a.kennzeichen,"A-BC 999");
    a.besitzer=&p;

    printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    strcpy(p.nachname,"Schulze");
    printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    return 0;
}
```

Initialisierung der **Strukturkomponenten** von a

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    int id;
    char nachname[31];
}
PERSON;

typedef struct
{
    char kennzeichen[31];
    PERSON* besitzer;
}
AUTO;

int main(void)
{
    PERSON p;
    p.id=1234;
    strcpy(p.nachname,"Mustermann");

    AUTO a;
    strcpy(a.kennzeichen,"A-BC 999");
    a.besitzer=&p;

    printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    strcpy(p.nachname,"Schulze");
    printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    return 0;
}
```

Insbesondere:
a.besitzer wird mit dem **Pointer** auf eine Variable vom Typ PERSON belegt

WBS TRAINING

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    int id;
    char nachname[31];
}
PERSON;

typedef struct
{
    char kennzeichen[31];
    PERSON* besitzer;
}
AUTO;

int main(void)
{
    PERSON p;
    p.id=1234;
    strcpy(p.nachname,"Mustermann");

    AUTO a;
    strcpy(a.kennzeichen,"A-BC 999");
    a.besitzer=&p;

    printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    strcpy(p.nachname,"Schulze");
    printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    return 0;
}
```

Kontrollausgaben aller Strukturkomponentenwerte
und
Überschreibung eines Strukturkomponentenwertes

WBS TRAINING

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    int id;
    char nachname[31];
}
PERSON;

typedef struct
{
    char kennzeichen[31];
    PERSON* besitzer;
}
AUTO;

int main(void)
{
    PERSON p;
    p.id=1234;
    strcpy(p.nachname,"Mustermann");

    AUTO a;
    strcpy(a.kennzeichen,"A-BC 999");
    a.besitzer=&p;

    printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    strcpy(p.nachname,"Schulze");
    printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    return 0;
}
```

Insbesondere:

mit **(*a.besitzer)** wird zunächst der Pointer dereferenziert, der als Strukturkomponente von a auf eine PERSON-Strukturvariable verweist.

mit **.nachname** wird dann bei dieser Variable die entsprechende Strukturkomponente angesprochen

WBS TRAINING

Struktur – **Beispielaufgabe (3)** – **Quellcode**

```
#include<stdio.h>
#include<string.h>

typedef struct
{
    int id;
    char nachname[31];
}
PERSON;

typedef struct
{
    char kennzeichen[31];
    PERSON* besitzer;
}
AUTO;

int main(void)
{
    PERSON p;
    p.id=1234;
    strcpy(p.nachname,"Mustermann");

    AUTO a;
    strcpy(a.kennzeichen,"A-BC 999");
    a.besitzer=&p;

    printf("Der Besitzer von %s hat aktuell den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    strcpy(p.nachname,"Schulze");
    printf("Der Besitzer von %s hat nun den Nachnamen %s\n",a.kennzeichen, (*a.besitzer).nachname);
    return 0;
}
```

```
Der Besitzer von A-BC 999 hat aktuell den Nachnamen Mustermann
Der Besitzer von A-BC 999 hat nun den Nachnamen Schulze
```

Die Kontrollausgaben weisen nach, dass der PERSONEN-Pointer innerhalb der Strukturvariable Auto tatsächlich auf eine externe Strukturvariable vom Typ PERSON verweist
=>
Wenn der Nachname des Besitzers überschrieben wird, so ist dies auch innerhalb der Auto-Variable ersichtlich.

Struktur – Abschließender Hinweis:

- Wir haben bisher alle Strukturen mit Hilfe von „**typedef**“ eingeführt.
- Darauf könnte aber auch verzichtet werden.
  (Gelegentlich wäre sogar ein sofortiger Einsatz von **typedef** unzulässig und könnte erst im Anschluss nachgeholt werden - siehe hierzu die heutige FPA)
- Der Vollständigkeit halber soll nun aber zum Abschluss noch gezeigt werden, welche Syntax zu verwenden ist, falls auf typedef **vollständig** verzichtet wird:

Schreibeise MIT typedef:

```
typedef struct
{
        char kennzeichen[31];
        int baujahr;
}
AUTO;

// Deklaration einer Strukturvariable a vom Typ AUTO
AUTO a;
```

Schreibeise OHNE typedef:

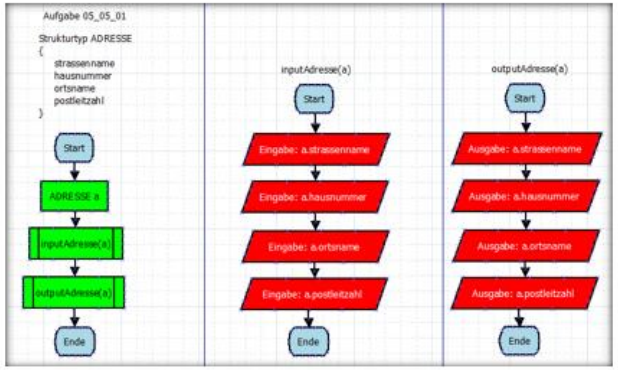
```
struct AUTO
{
        char kennzeichen[31];
        int baujahr;
};


// Deklaration einer Strukturvariable a vom Typ AUTO
struct AUTO a;
```

Struktur – Gemeinsame Übung A_05_05_01

WBS TRAINING

Aufgabe_05_05_01

Gegeben sei der folgende PAP:



Aufgabenstellung:
Bitte erstellen Sie dazu einen geeigneten **Quellcode** in ANSI C.

WBS TRAINING

VIELEN DANK FÜR IHRE AUFMERKSAMKEIT!