# Programmiertechnik IKA 9/13 – Übungsblatt 12.03.2014

## Aufgabe 1 (Strukturen):

Deklarieren Sie eine Struktur „Schueler“ mit den Eigenschaften:

- Matrikelnummer (int)
- Vorname (string)
- Nachname (string) und
- Note (float)

Definieren Sie nun zwei (2) Schüler IKAler1 und IKAler2, einmal mit einer separaten Deklaration und dem Zuweisen der 4 Werte in jeweils einer eigenen Zeile – und einmal mit sofortiger Initialisierung = {…}

Geben Sie die beiden Schüler und deren Eigenschaften anschließend am Bildschirm aus.

## Aufgabe 2 (Datenfeld aus Strukturen):

Verwenden Sie die Struktur aus Aufgabe 1 und legen Sie nun ein Datenfeld **IKA** für 10 Schüler an. Weisen Sie zumindest den ersten beiden Schülern passende Werte zu und geben Sie alle Schüler in einer Schleife untereinander tabellarisch aus.

## Aufgabe 3 (Strukturen):

Deklarieren Sie eine Struktur „Kunde“ mit den Eigenschaften:

- „Kundennummer“ (eine Ganzzahl im Bereich von 100000 bis 999999) und
- „Kundenname“ (eine string-Zeichenkette)

Legen Sie zwei Kunden an (Kunde1 und Kunde2) und „verpassen“ Sie den Kunden passende Werte für diese Eigenschaften.

Zeigen Sie die beiden Kunden am Bildschirm an.

## Aufgabe 4 (Datenfeld aus Strukturen):

Verwenden Sie die Struktur aus Aufgabe 3 und definieren Sie ein Datenfeld für 5 Kunden – mit sofortiger Initialisierung…

Der Benutzer soll nun eine Kundennummer eingeben, das Programm durchsucht daraufhin das Datenfeld nach genau dieser Kundenummer.

Wurde der Datensatz gefunden, gibt das Programm den dazu gehörenden Kundennamen aus, wenn nicht, soll eine entsprechende Fehlermeldung ausgegeben werden.

**Erweiterungsmöglichkeit (freiwllig):**

Das Programm fragt solange nach einer zu suchenden Kundennummer, bis der Benutzer die Zahl 0 eingibt. Etwa so:

```
Welche Kundennummer suchen Sie: (Ende mit 0) 232323
Kundennummer 232323 leider nicht gefunden...

Welche Kundennummer suchen Sie: (Ende mit 0) 345678
Kundennummer 345678 gehoert zu Fernando Alonso

Welche Kundennummer suchen Sie: (Ende mit 0) 0
Drücken Sie eine beliebige Taste . . .
```

## Aufgabe 5 (Datenfeld von Strukturen):

Sie erinnern sich sicher unserer Dönerbude!?!?

Wir haben verschiedene Speisen angeboten, Bestellungen entgegen genommen und den Gesamtpreis ausgerechnet, etwa so:

```
Willkommen in unserer Doenerbude!

Wir koennen Ihnen heute 3 Speisen anbieten:
1. Lahmacun            Preis: 3.75 EUR
2. Doener-Kebap        Preis: 3.45 EUR
3. Pommes-Frites       Preis: 2.50 EUR

Wieviele Lahmacun moechten Sie kaufen?  2
Wieviele Doener-Kebap moechten Sie kaufen?  1
Wieviele Pommes-Frites moechten Sie kaufen?  2

Besten Dank fuer Ihre Bestellung, die wir wie folgt ausfuehren:
2 * Lahmacun,      kostet zusammen 7.50 EUR
1 * Doener-Kebap,  kostet zusammen 3.45 EUR
2 * Pommes-Frites, kostet zusammen 5.00 EUR

Gesamtwert Ihrer Bestellung: 15.95 EUR. Guten Appetit!

Drücken Sie eine beliebige Taste . . .
```

Lösen Sie die Aufgabe nun unter Verwendung von einer (1) Struktur namens „**Angebot**“, diese Struktur enthält den Namen der angebotenen Speise und natürlich den Einzelpreis.

**HINWEISE:**

- Verwenden Sie ein Datenfeld für 3 Strukturvariablen.
- Verwenden Sie jeweils eine Schleife für die Ausgabe der 3 Speisen, im Beispiel jeweils kursiv dargestellt.
- **TIPP:** Legen Sie in der Struktur zusätzlich eine Variable für die Anzahl der bestellten Speisen und eine Variable für den Summenpreis der bestellten Speise an.

Überprüfen und gegebenenfalls korrigieren Sie Ihr Programm so, dass Sie NUR in der Definition Ihrer Angebote zusätzliche „Cheeseburger-Spezial (2.45 €)“ und/oder „Hawaiitoast (3.95 €)“ anbieten können, ohne dass der Rest des Programms angepasst werden müsste…

„NUR“ in der Definition – nun ja, Sie werden, wenn Sie es bisher noch nicht vorgesehen haben, eine (1) zusätzliche Variable für die Anzahl der angebotenen Speisen benötigen.

## Aufgabe 6 (typedef):

Deklarieren Sie einen eigenen Datentyp „const unsigned short int“, legen Sie eine Variable diesen Datentyps und weisen ihr einen negativen Wert zu. Geben Sie den Wert aus.

## Aufgabe 7 (enum):

Legen Sie 2 Enumerationstypen an wie im Unterricht gezeigt:

**`Lehrgang`** und **`Obst`**

Legen Sie für jeden Enum-Typ eine Variable an und weisen ihr (ohne Benutzereingabe) Werte zu, das heißt: probieren Sie verschiedene Wertzuweisungen, kompilieren Sie jedesmal neu und schauen, welche zugewiesenen Werte vom Compiler akzeptiert werden und welche nicht.