Deklaration von Strukturen

1. Begriffsdefinition 'Struktur'

Der Datentyp **`struct`** verbindet Variablen gleicher oder verschiedener Datentypen zu einem intern strukturierten Datentyp. Die Struktur wird durch einen Bezeichner, z.B. **`S`** adressiert. **`S`** hat dann mit dem Punktoperator **`'.'`** Zugriff auf die Felder (*Attribute, Komponenten*). Der Datentyp *Struktur* kann *explizit* deklariert werden.

2. Explizite Deklaration

Soll beispielweise **`S`** als Struktur deklariert werden, erfolgt eine entsprechende *explizite* Deklaration durch

```
S = struct
```

Die Struktur **`S`** verfügt noch nicht über Felder (*Attribute, Komponenten*) und auch nicht über Feldwerte (*Attributwerte*). Soll die Struktur über Komponenten, z.B. **`feld1, feldN`** gegeben werden, können diese Felder (*Attribute*, *Komponenten*) dann durch die Anweisungen

```
S.feld1
...
S.feldN
```

hinzugefügt werden. Eine entsprechende Befehlssequenz kann wie folgt aussehen:

Beispiel 1

```
>> Person = struct
>> Person.name = ''
>> Person.vname = ''
>> Person.telefon = ''


>> Person
Person =
          name : ''
         vname : ''
       telefon : ''
```

3. Begriffsdefinition 'Strukturarray'

Der Datentyp **[struct]** verbindet Exemplare (*Strukturvariable*) einer Struktur zu einem indizierten Variablen Strukturarray. Die Struktur wird durch einen Bezeichner **S(Index)** adressiert. Ist der Index **k**, dann hat **S(k)** mit dem Punktoperator **'.'** Zugriff auf die Felder (*Attribute, Komponenten*) der **k**−ten Strukturvariablen (Exemplars von **[S].** Der Indexbereich von **S** ist

{1 <= k <= N}.

4. Explizite Deklaration

Soll beispielweise **S** als Strukturarray deklariert werden, erfolgt eine entsprechende explizite Deklaration durch

```
S = [struct]
```

Die Struktur **S(k)** verfügt noch nicht über Felder (*Attribute, Komponenten*) und nicht über Feldwerte (*Attributwerte*). Sollen der Struktur **S(k)** die Felder (*Attribute, Komponenten*) **feld1, ..., feldN** zugeordnet werden, werden diese Felder (*Attribute*, *Komponenten*) durch die Sequenz

```
<Bezeichner> (<Indexwert>).<Feldname> = <Attributwert>
...
```

deklariert und ihnen der Attributwert **<Attributwert>** zugewiesen. Eine Strukturfelddeklaration, die den Komponenten den String '' zuweist, ist wie folgt aufgebaut [1]

```
S(<Indexwert>).feld1 = ''
...
S(<Indexwert>).feldN = ''
```

und kann z.B. durch folgende Befehlssequenz realisiert werden:

Beispiel 2

```
>> Person = [struct]
```

[1] Die Metanotation ***<Indexwert>*** bedeutet, daß der Term ***Indexwert*** durch einen Wert aus dem Wertebereich von k zu ersetzen ist.

```
>> Person(1).nname = ''
>> Person(1).vname = ''
>> Person(1).telefon = ''


>> Person(1)

ans =
          name    : ''
         vname    : ''
       telefon    : ''
```

5. Wertzuweisung

Besitzt eine Struktur Felder (*Attribute*, *Komponenten*), etwa die Komponenten

<feldname> → **feld1, ... feldN**,

dann kann ihnen durch die Sequenz

<Bezeichner>(<Indexwert>).<feldname> = <Attributwert>

ein Feldwert **<Attributwert>** zugewiesen werden.

Beispiel 3

```
>> Person = [struct]
>> Person (1).nname = 'meier'
>> Person (1).vname = 'max'
>> Person (1).saldo = -2.37
>> Person (1).telefon = 6563113
>> Person (2).nname = 'müller'
>> Person (2).vname = 'otto'
>> Person (2).saldo = -12,99
>> Person (2).telefon = 1237654

>>Person (2)
ans =
    name : 'müller'
   vname : 'otto'
   saldo : -12.99
telefon : 1237654
```

Der Zugriff auf die hier definierte Strukturarray erfolgt vorteilhaft durch ein Schleifenkonstrukt. Dies kann wie folgt mit einer for-Schleife und einem Index realisiert werden::

Beispiel 4

```
for (idx = 1: 1: 2)

     fprintf("Name= %s\n", S(idx).name);
     fprintf("Vname = %s\n", S(idx).vname);
     fprintf("Saldo = %s\n", S(idx).saldo);
```

```
    fprintf("Telefon = %s\n", S(idx).plz);
end
```

6. Wichtige MATLAB-Funktionen

MATLAB stellt einige eingebaute Funktionen zur Manipulation von Strukturen bereit.

6.1 setfields ()

Die Funktion **setfields()** weist Feldinhalte (*Attributwerte*) zu. Der Funktionsaufruf

```
setfields ('Person', 'wohnort', 'Luebeck')
```

fügt der Struktur **Person** das Feld mit der Bezeichnung **wohnort** mit dem Attributwert '**Luebeck**' zu.

6.2 getfields ()

Die Funktion **getfields ()** ermittelt Feldinhalte (*Attributwerte*). Der Funktionsaufruf

```
w = getfields ('Person', 'wohnort')
```

ermittelt den Attributwert des Feldes **wohnort** der Struktur **Person** und weist ihn als Rückgabewert der Variablen **w** zu.

6.3 rmfields ()

Die Funktion **rmfields ()** löscht ein Feld aus einer Struktur. Der Funktionsaufruf

```
rmfields ('Person', 'wohnort')
```

löscht das Datenfeld **wohnort** aus der Struktur **Person.** Da **rmfields ()** nicht auf der Struktur, sondern nur auf einer Kopie der Struktur operiert, muß das Resultat ggf. durch

```
Person = rmfields (Person, 'wohnort')
```

erneut zugewiesen werden

6.4 orderfields ()

Die Funktion **orderfields ()** ordnet die Felder einer Struktur aufsteigend in alphabetischer Reihenfolge. Der Funktionsaufruf

```
orderfields (Person)
```

ordnet die Felder der Struktur **Person** aufsteigend.