

# ***Grundlagen der Theoretischen Informatik***

Sebastian Iwanowski  
FH Wedel

## **Kap. 3: Verifikationstechniken**

### **Teil 4: Modularisierung und funktionale Programmierung**


Vorbereitungsmaterial zum Selbststudium  
(gehört thematisch zu Programmiersprachen 1)

# Modularisierung

**Grundlegende Idee: Zerlege ein Problem in Teilprobleme**

**1. Beispiel:** Schreib einen Brief

Briefkopf;  
Anrede;  
Briefinhalt;  
Briefschluss



**Prozeduren**

***Die 4 Teilprobleme können unabhängig voneinander gelöst werden und wiederverwendet werden.***

**Vorteile:**

- 1) Übersichtlichkeit**
- 2) Wiederverwendbarkeit**

# Modularisierung

## Grundlegende Idee: Zerlege ein Problem in Teilprobleme

**2. Beispiel:** Berechne zu einem Kalenderdatum  $D$  und einer Tageszahl  $n$  das Kalenderdatum  $D+n$

```
LiesEingabedatum;  
LiesTageszahl;  
BerechneAusgabedatum;  
GibAusgabedatumAus;
```

**Problem:** Hier müssen Daten zwischen den Prozeduren ausgetauscht werden.

**Lösung:** Prozeduren mit Parametern

```
LiesEingabedatum (in);  
LiesTageszahl (n);  
BerechneAusgabedatum (in, n, out);  
GibAusgabedatumAus (out);
```

# Modularisierung

## Prozeduren mit Parametern

```
LiesEingabedatum (in);  
LiesTageszahl (n);  
BerechneAusgabedatum (in, n, out);  
GibAusgabedatumAus (out);
```

### Eingabeparameter:

- dienen der Übermittlung von Werten aus dem aufrufenden Programm an die Prozedur

### Ausgabeparameter:

- dienen der Übermittlung von Werten aus der Prozedur an das aufrufende Programm

### 1. Übergabetechnik: Call by reference (Variablenparameter)

- Prozedur und aufrufendes Programm teilen sich denselben Speicherplatz.
- In Pascal wird diese Übergabetechnik bei den VAR-Parametern angewandt.
- In manchen Programmiersprachen gibt es nur diese Übergabetechnik.

# Modularisierung

## Prozeduren mit Parametern

```
LiesEingabedatum (in);  
LiesTageszahl (n);  
BerechneAusgabedatum (in, n, out);  
GibAusgabedatumAus (out);
```

### Eingabeparameter:

- dienen der Übermittlung von Werten aus dem aufrufenden Programm an die Prozedur

### 2. Übergabetechnik: Call by value (Wertparameter)

- Prozedur und aufrufendes Programm legen für denselben Parameter verschiedene Speicherplätze an.
- Beim Aufruf wird der Wert des aufrufenden Programms in den entsprechenden Speicherplatz der Prozedur kopiert.
- Eine Rückgabe von Werten durch die Prozedur ist auf diese Weise **nicht** möglich (also nur als Eingabeparameter verwendbar).
- In Pascal wird diese Übergabetechnik bei den Nicht-VAR-Parametern angewandt.
- In manchen Programmiersprachen gibt es nur diese Übergabetechnik.

# Modularisierung

## Parameteridentifikation zwischen aufrufendem Programm und Prozedur

im aufrufenden Programm:

*Aktuelle Parameter*

```
BerechneAusgabedatum (in, n, out);
```

*Wertübergabe bzw.  
Adressübergabe  
(je nach verwendeter  
Übergabetechnik)*

in der Prozedur:

```
BerechneAusgabedatum (inDate, count, outDate);
```

*Formale Parameter*

- Es werden jeweils die Parameter identifiziert, die an der gleichen Position in der Parameterliste stehen.

# Modularisierung

## Anforderungen an die Parameter

### an die formalen Parameter (in der Prozedur):

- Parameter müssen Variablennamen sein.
- Bei Call by reference steht dieser Variablenname für die Speicheradresse, die übergeben wurde.
- Bei Call by value steht dieser Variablenname für einen in der Prozedur neu eingerichteten Speicherplatz, in den der Wert des aufrufenden Programms kopiert wird.

### an die aktuellen Parameter (im aufrufenden Programm):

- Bei **Call by reference** müssen die Parameter **Variablennamen** sein. Sie stehen für die Speicheradresse, die an die Prozedur übergeben wird.
- Bei **Call by value** dürfen die Parameter **beliebige Wertausdrücke** sein. Der Ausdruck muss vor Prozedurbeginn ausgewertet werden und wird dann in den neu eingerichteten Speicherplatz der Prozedur kopiert.