ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Methoden
Prozeduren, Funktionen, globale/lokale Variablen

Programmieren und Problemlösen

Prof. Hans Hinterberger
Barbara Scheuner

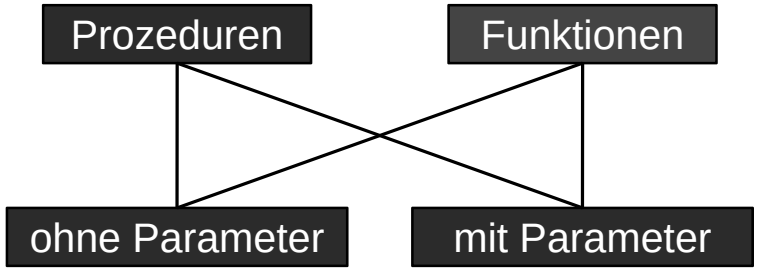
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Ablauf

- Kleines Beispiel
  - Grundgerüst
  - Aufruf
- Lokale / globale Variablen
- Methoden
  - Prozeduren
  - Funktionen
- Aufrufen von Methoden
- Sichtbarkeit von Variablen (Wiederholung)

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Übersicht



ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Aufbau einer Prozedur

```
PROCEDURE name;
BEGIN
  {Anweisungen}
END;
```

```
PROCEDURE writeHallo;
BEGIN
  writeln('hallo');
END;
```

Beispiel

```
PROGRAM myprog;
VAR
{variablendeklaration}

    PROCEDURE myproc;
    BEGIN
        {Anweisungsblock}
    END;

BEGIN
    {Anweisungen}
     myproc;
    {Anweisungen}
END.
```

```
PROGRAM myprog;
VAR
    globalInt: Integer;

    PROCEDURE writeHallo;
    BEGIN
        writeln('hallo');
    END;

→ BEGIN
    writeHallo;
END.
```

Beispiel: Globale vs. Lokale Variablen

```
PROGRAM myprog;
VAR
{variablendeklaration}

    PROCEDURE myproc;
    VAR
        {Variablendeklaration}
    BEGIN
        {Anweisungsblock}
    END;

BEGIN
    {Anweisungen}
     myproc;
    {Anweisungen}
END.
```

```
PROGRAM myprog;
VAR
    globalInt: Integer;

    PROCEDURE myproc;
    VAR
        localInt: Integer;
    BEGIN
        localInt:= 2;
        globalInt:= localInt +2;
    END;

→ BEGIN
    myproc;
    globalInt:= 7;
END.
```

Fragen:
- Welchen Wert hat `GlobalInt` ?
- Welchen Wert hat `LocalInt`?

Parameter

- Parameter werden von der aufgerufenen Prozedur zur „Bewältigung“ ihrer Aufgaben benötigt.
- Parameter haben einen Typ. Die Prozedur kann nur mit Parametern des richtigen Typs aufgerufen werden.
- Warum Parameter?
  - Einfacher Parameter: Variablen-Kopie
  - VAR-Parameter: “richige” Variable

Prozeduren

- Ohne Parameter:
  `procedure` **`readln;`**
- Mit Parametern:
  `procedure` **`readln`**`(text: String);`
- Mit var Parametern:
  `procedure` **`val`**`(text: String, var wert: Integer, var error: Integer);`

Parameter

Die Prozedur:

```
procedure Showmessage(text: String);
```

Aufruf mit einer *Konstanten*:

```
Showmessage(´dies ist ein Parameter´);
```

Aufruf mit einer *Variablen*:

```
text:= ´dies ist ein Parameter‘;
Showmessage(text);
```

Parameter Beispiel

```
PROCEDURE printInt(nummer: Integer);
VAR
  text: String;
BEGIN
  text:= StrtoInt(nummer);
  Showmessage(text);
END;
```

```
printInt(7);
printInt(wert);
```

Variable Parameter

Die Prozedur:

```
PROCEDURE Change(var text: String);
BEGIN
    text:= ´dies ist eine Variable ´;
END;
```

Aufruf mit einer *Konstanten* nicht möglich

Aufruf mit einer *Variablen*:

```
text:= ´dies ist ein Parameter´;
Change(text);
writeln(text);
```

var-Parameter Beispiel

```
PROCEDURE printInt(nummer: Integer,
                   var text: String);
Begin
  text:= StrtoInt(nummer);
  Showmessage(text);
End;
```

```
printInt(7, ´7´);
printInt(7, text);
```

Funktionen

- Funktionen haben **einen** Typ (Rückgabewert)
- Funktion ohne Parameter:
  `function getCurTime(): Integer;`
- Funktion mit Parameter:
  `function StrToInt(text: String): Integer;`
- Funktion mit var Parameter:
  `function isNumber(var text: String): Boolean;`

Beispiel (1)

```
Function abc(nummer: Integer): String;
var
   text: String;
   i: Integer;
begin
   text:= ´´;
   for i:= 0 to nummer do begin
       text:= text + chr(65+i);
   end;
   abc:= text;
end;
```

```
Text:= abc(7);
```

Verwendung von Funktionen

- Das Resultat eines Funktionsaufrufes kann ohne Zwischenspeicherung verwendet werden.

Beispiele:

```
Function sqrt(wert: Real): Real;
x:= 3 + sqrt(7);

Function test(text: String): Boolean;
IF test('hallo') THEN …
```

Methoden

- Methoden = Prozeduren und Funktionen
- Methoden dienen der besseren Strukturierung des Programms.
- Sich wiederholende Programmabschnitte können in eine Methode geschrieben werden.
  -> Mehrfachverwendung
  Z.B. IntToStr()

```
function IntToStr(Value: Integer): string;
```

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Parameter vs. Globale Variablen

- Globale Variablen:
  - Alle Variablen sind an einem Ort
  - Alle Methoden haben darauf Zugriff
    - Eventuell Verlust der Kontrolle
  - Bei vielen Variablen unübersichtlich
- Parameter:
  - Die Sichtbarkeit ist klar geregelt
  - Übersichtlichkeit
  - Der Speicherplatz kann wieder freigegeben werden

```
PROGRAM pandemie;
USES crt;
VAR
   personen: array[0..21,0..51] OF INTEGER;
   i,j, tage: Integer;
BEGIN
   FOR i:= 0 to 21 DO BEGIN
      FOR j:= 0 to 51 DO BEGIN
         personen[i,j]:= 0;
      END;
   END;
   setztekranke;
   print;
   FOR tage:= 1 TO 10 DO BEGIN
      FOR i:= 1 to 20 DO BEGIN
         FOR j:= 1 TO 50 DO BEGIN
            ansteckung(i,j);
         END;
      END;
      clrscr;
      writeln('Tag ',tage);
      print;
   END;
   readln;
END.
```

Alle Werte im 2-dim Array personen auf 0 setzen

Alle Personen anstecken

Simulation der Pandemie für 10 Tage