Implementierung einer (einfachen) Programmiersprache

François Bry

Heutige Vorlesung

1. Rechnen mit einem Keller
2. Variablen
3. Wiederholungsschleife
4. Einfache Prozeduren
5. Verbesserung des Prozedurkonzeptes
6. Schlussbemerkungen

1. Rechnen mit einem Keller

1.1 Postfix-Notation

1.2 Keller zum Rechnen

1.3 Rechnen mit Boole'schen Werten

1.4 Abstraktion

2. Variablen

Berechnung des arithmetischen Durchschnitts zweier Werte:

```
var Wert1;
var Wert2;
var arithmetischerDurchschnitt;
arithmetischerDurchschnitt
                        := (Wert1 + Wert2) / 2;
```

2. Variablen

Vertauschen der Werte zweier Variablen:

```
var Var1;
var Var2;
var Hilfsvariable;
Hilfsvariable := Var1;
Var1 := Var2;
Var2 := Hilfsvariable;
```

2. Variablen

Eine weitere Berechnung:

```
var x;
var y;
x := (1 + 2) * 3;
y := 2;
x := x + y;
```

2.3 Darstellung von Variablen auf dem Keller

Symboltabelle

Befehle:

- reset
- inc 1
- pushlit
- store 0 1
- op *

2.4 Benannte Konstanten

```
const Steuersatz := 25%;
const Kindergeld := 1848;
var Einnahmen;
var Kinderzahl;
var absetzbareKosten;
var Steuerschuld;
var kindesbezogeneSteuerentlastung;

kindesbezogeneSteuerentlastung
        := (Kinderzahl * Kindergeld * (100 - Steuersatz));
 Steuerschuld := (((Einnahmen - absetzbareKosten) *
        Steuersatz) - kindesbezogeneSteuerentlastung);
```

2.5 Typ einer Variable oder Konstante

```
const real Steuersatz := 0.25;
var integer Kinderzahl;
var boolean steuerpflichtig := true;
```

3. Wiederholungsschleife

```
var Kapital;
var Zinssatz;     (* Dezimalwert *)
var Dauer;
var i := 1;          (* Schleifenindex *)

read Kapital;
read Dauer;    (* nat. Zahl >= 1 *)
while (i =< Dauer)
    { Kapital := Kapital * (1 + Zinsatz) ; i := i + 1; }
write Kapital;
```

Übersetzung

```
0: inc 1
1: inc 1
2: inc 1
3: inc 1
4: pushlit 1
5: store 0 3
6: read
7: store 0 0
8: read
9: store 0 2
10: goto 21
11: push 0 0
12: pushlit 1
13: push 0 1
14: op +
15: op *
16: store 0 0
17: push 0 3
18: pushlit 1
19: op +
20: store 0 3
21: push 0 3
22: push 0 2
23: op =<
24: jumpontrue 11
25: write 0 0
```

4. Einfache Prozeduren

```
program Schuldenrechner
  { var Schuld;
    var Zinssatz;
    var Dauer;
    procedure
          Schuldberechnung
      { var Zinsen := 0;
       procedure
          Zinsenberechnung
        { var Index := 1;
          while (Index
            =< Dauer)
       { Zinsen := (Schuld *
    Zinssatz);
                Index := Index+1; } }
      call  Zinsenberechnung;
      Schuld := Schuld +
    Zinsen; }
  read Schuld;
  read Zinssatz;
  read Dauer;
  call   Schuldberechnung;
  write Schuld; }
```

```
0: reset
1: inc 1
2: inc 1
3: inc 1
4: goto 30

5: inc 1
6: goto 24

 7: inc 1
 8: pushlit 1
 9: store 0 0
10: goto 19
11: push 2 0
12: push 2 1
13: op *
14: store 1 0
15: push 0 0
16: pushlit 1
17: op +
18: store 0 0

19: push 0 0
20: push 2 2
21: op =<
22: jumpontrue
                13
23: return
24: call 8
25: push 1 0
26: push 0 0
27: op +
28: store 1 0
29: return

30: read 0 0
31: read 0 1
32: read 0 2
33: call 5
34: write 0 0
35: halt
```

5. Verbesserung des Prozedurkonzeptes

5.1 Aufrufparameter

5.2 Funktionen

5.3 Rekursion

5.1 Aufrufparameter

```
program Schuldenrechner
  { var Schuld;
    var Zinssatz;
    var Dauer;

    procedure Schuldberechnung
                                  (S, ZS, D)
      { var Zinsen := 0;
        procedure Zinsenberechnung
                              (S1, ZS1, D1, Z1)
          { var Index := 1;
            while
            (Index=<D1)
                { Z1 := (S1 * ZS1);
                  Index:=Index+1;} };
          call Zinsenberechnung
                              (Zinsen, ZS, D);
          S:=S + Zinsen; }
read Schuld;
read Zinsatz;
read Dauer;
call Schuldberechnung
            (Schuld, Zinssatz, Dauer);
write Schuld; }
```

5.2 Funktionen

```
Schuld := Schuld
    + Zinsenberechnung(Schuld, Zinssatz, Dauer );
```

5.3 Rekursion

```
Fakultaet(n) := if n = 1
                    then 1
                    else n * Fakultaet(n - 1);
```

Schlussbemerkungen (1)

- Es gibt viele Programmierparadigmen
- Es gibt noch mehr Programmiersprachen
- Es werden noch viele kommen

Schlussbemerkungen (2)

- Ein Programm ist für Menschen, nicht für Computer.
- Wer nicht programmiert, kann kein fähiger Informatiker sein.
- Eleganz ist in der Programmierung nicht optional.
- Man programmiert erst dann gut, wenn man weiß, wie die Sprache implementiert ist.