7. Übungsblatt zu Programmiersprachenkonzepte

Manuel Schwarz, Michael Stypa

8. Dezember 2010

Aufgabe 7.1

Scope

- Anweisungsbreich, in dem eine Variable sichtbar ist (globale vs. lokale Variable)
- static scope → Scope zur Compilezeit
- dynamic scope → Scope erst zur Laufzeit bestimmbar

Lebenszeit

- $\neq$ scope
- Zeitraum in dem eine Variable Speicherplatz belegt
- Lebensdauer lokaler Variablen: vom Funktionsaufruf bis Verlassen der Funktion
- Lebensdauer globaler Variablen: ganze Programmdauer
- statische lokale Variablen (z.B. C): Lebensdauer wie globale, Gültigkeitsbereich: nur lokale Funktion

Aufgabe 7.2

Statisches Binden

- Binden zur Compilezeit
- bleibt bis zum Programmende bestehen
- alle benötigten Daten(-typen) werden mit in die ausführbare Datei gebunden
- beschleunigt die Ausführung des Programms, führt aber zu mehr Platzverbrauch

Dynamisches Binden

- Binden zur Laufzeit
- kann sich während der Programmausführung ändern
- Daten werden erst bei Bedarf (zur Laufzeit) nachgeladen
- weniger Platzbedarf, aber Typ-Bindung und -Überprüfung zur Laufzeit kostet viel Rechenaufwand
- Sprachen mit dynamischer Typbindung haben i.d.R. Interpreter

Aufgabe 7.3

Bezeichner bestehen aus 1 bis 127 signifikanten Characters.

```
program identifier;
const
    aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
    aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
    aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
begin
end.
```

Aufgabe 7.4

```
program aufg4;
const
    x51 = 'b';
    x52 = 20;
    x7  = 1;
    x8  = 10;
    x10 = 'C';
var
    x1  : boolean;
    x2  : integer;
    x3  : real;
    x41 : char;
    x42 : integer;
    x6  : boolean;
    x9  : real;
    x11 : real;
    x12 : real;
begin
    x1  := odd(x2) < (sqrt(x3) >= 3.8);
    x41 := pred(x51);
    x42 := pred(x52);
```

```
    x6  := chr(sqr(abs(x7 * x8))) <> 'H';
    x9  := ord(x10) * x11 / x12
end.
```

Aufgabe 7.5

```
1  program aufg3();
2  const
3      min=20; max=10;
4      grad1=7.5e; grad2=+2.5e+2;
5      minuspi=-pi;
6      pi=3.14;
7      ja=false;
8  var
9      min:Integer;
10     zeichen:char;
11     wahr:bool;
12     faktor, sin:real;
13 begin
14     for i=min to max do
15     begin
16         wahr := ja; sin := 10;
17         while wahr do
18             factor := sin DIV pi;
19             if (factor > min) then wahr := false;
20                               else sin := sqr(sin);
21     end;
22     writeln('faktor ist ',factor);
23     writeln(pi);writeln(minuspi);
24 end;
```

Korrektur:

Zeile 1 Klammern weg (oder Parameter rein)

Zeile 3 In dieser Reihenfolge startet die `for` -Schleife nicht

Zeile 4 `7.5e+0`

Zeile 5 Deklaration von `pi` vor der von `minuspi`
`pi` ist Systemvariable und wird überschrieben

Zeile 9 Variablen- versucht Konstatendeklaration zu überschreiben.

Zeile 11 `boolean`

Zeile 12 `factor`

Zeile 17 Körper mit `begin` und `end` für die `while` -Schleife

Zeile 18 `/` für real

Zeile 19 kein `;` vor `else`