Übungsblatt 9 zu "Programmiersprachen"

Berthold Hoffmann, Studiengang Informatik (hof@informatik.uni-bremen.de)

Besprechung am 18. Mai 2010

call-by-name

Betrachten Sie folgendes Programm in der prähistorischen Sprache Algol-60:

```
begin
  integer i;
  integer g;
  real array poly[0:g];
  real y, f of y;
  real procedure horner (coeff,x);
    value x;
    real coeff;
    real x;
    begin comment Horner scheme for polynamial functions;
      real h;
      i:= n;
      h:= coeff;
      for i:= n−1 step −1 until 0
      do h:= h ∗ x + coeff;
      horner:= h
    end horner;

  g:=3;
y:= 2.0;
  poly[3]:= 3.0;
  poly[2]:= 2.0;
  poly[1]:= 1.0;
  poly[0]:= 10.2;
  print(horner(poly[i], y))
end
```

Zunächst einige Anmerkungen zur Syntax von Algol-60:

1. In Prozeduren werden die Typen der formalen Parameter zwischen dem Kopf "**procedure** $p(f_1, \ldots, f_n)$;" und dem Rumpf "**begin** ... **end**" definiert. Dabei zeichnet "value f_i;" den Parameter f_i als Wertparameter aus – alle anderen werden mit *call-by-name* übergeben.

2. Im Rumpf einer Prozedur p definieren Zuweisungen $p := \langle E \rangle$ den Ergebniswert einer Funktionsprozedur.

Fragen:

1. Welcher Befehl wird ausgeführt, wenn der Aufruf "horner(poly[i], y)" ausgeführt wird?
2. Welchen Wert gibt das Programm aus?
3. Was passierte, wenn im Rumpf der Prozedur eine Vereinbarung "**integer** i;" eingefügt würde? (Algol-60 hat *statische Bindung*!)
4. Welche Probleme können Programmierer mir *call-by-name* bekommen?