PHILIPPS-UNIVERSITÄT MARBURG D-35032 Marburg
Fachbereich Mathematik und Informatik Hans Meerwein Straße
Prof. Dr. R. Loogen
J.Berthold

4.Juli 2006

11. Übung zu "Semantik von Programmiersprachen", SS 2006

Abgabe schriftlicher Aufgaben: Di, 11.Juli 2006 (vor der Vorlesung)
Besprechung mündlicher Aufg.: 6.Juli 2006

Hinweise: Dies ist das letzte Übungsblatt mit schriftlichen Aufgaben. Insgesamt sind in den schriftlichen Übungen 135 Punkte erreichbar.

Mündliche Aufgaben

11.1 **Prozeduren und Schleifen**
Bekanntlich kann jede **while**-Schleife durch eine passende (endrekursive) Prozedur simuliert werden. Weisen Sie dies formal nach: Zeigen Sie für beliebige $b \in \mathbf{BExp}$ und $c \in \mathbf{Cmd}$[a]: $\mathcal{C}[\![\ \textbf{while}\ b\ \textbf{do}\ c]\!] = \mathcal{C}[\![P]\!]$,
wobei $P \equiv$ **begin**
proc p **is** (**if** b **then** (c; **call** p) **else skip**);
call p
end

[a] Ist es ein Problem, wenn in c ein Aufruf **call** p vorkommt? Was wäre die Konsequenz?

11.2 **Rekursive Fakultät**
Gegeben sei die nebenstehende Deklaration einer Prozedur zur Fakultätsberechnung. In der Startumgebung $\rho \in \mathbf{Env}$ sei $\rho(X) = v_x$ und $\rho(Y) = v_Y$.
Bestimmen Sie die durch die Deklarationssemantik bestimmte Umgebung.

```
proc p is
 begin  var Z;
        Z := X;
        if X = 1 then skip
        else ( X := X − 1;
               call p;
               Y := Y ∗ Z  )
 end
```

Schriftliche Aufgaben

11.3 **Schwächste Vorbedingung** 7 Punkte
Die Anweisung $c \equiv \textbf{while}\ \neg(X = 1)\ \textbf{do}\ (Y := Y{*}X; X := X{-}1)$ besitzt bekanntlich die folgende Semantik:

$$\mathcal{C}[\![c]\!]\sigma = \begin{cases} \sigma[Y \mapsto \sigma(Y) \cdot \sigma(X)!,\ X \mapsto 1] & : \ \sigma(X) > 0 \\ \bot & : \ \sigma(X) \leq 0 \end{cases}$$

Wir betrachten die Nachbedingung $B \equiv\ Y = i! \wedge i \geq 0$.

(a) Begründen Sie: $(X = i \wedge Y = 1)^I \neq \mathbf{wp}^I[\![c,\ B]\!]$ / 2

Bitte wenden!

(b) Finden Sie eine Zusicherung C, welche äquivalent zur schwächsten Vorbedingung der Zusicherung ist. Beweisen Sie dazu: / 5

$$\forall I \in \mathcal{I}: \quad C^I = \mathbf{wp}^I[\![c,\ B]\!].$$

11.4 **Totale Korrektheitsaussagen** 5 Punkte

Das Prinzip der axiomatischen Semantik lässt sich erweitern, um auch Aussagen über die Termination eines Programms zu machen.

Seien $P, Q \in$ **Assn** , $c \in$ **Cmd** . Wir definieren:

- Eine Aussage der Form $\{P\}\ c\ \{\Downarrow Q\}$ heißt *totale Korrektheitsaussage*.
- $\{P\}\ c\ \{\Downarrow Q\}$ heißt *gültig* , falls für alle $\sigma \in \Sigma$ und alle $I \in \mathcal{I}$ gilt:

$$\sigma \models^I P \curvearrowright \exists \sigma' \in \Sigma : \langle c, \sigma \rangle \to \sigma' \wedge \sigma' \models^I Q$$

- Zur Herleitung von totalen Korrektheitsaussagen wird eine Hoare-Regel für **while** definiert, in der die Schleifeninvariante P mit einer logischen Variablen $n \in \mathbb{N}$ parametrisiert ist.

$$\frac{\models P(n+1) \Rightarrow b \qquad \{P(n+1)\}\ c\ \{\Downarrow P(n)\} \qquad \models P(0) \Rightarrow \neg b}{\{\exists n.P(n)\}\ \ \mathbf{while}\ b\ \mathbf{do}\ c\ \{\Downarrow P(0)\}}$$

  Die übrigen Hoare-Regeln werden für totale Korrektheitsaussagen lediglich mit dem Zeichen $\Downarrow$ in der Nachbedingung ergänzt.

Formulieren Sie totale Korrektheitsaussagen für die folgenden Schleifen im Sinne dieser Definition und geben Sie geeignete parametrisierte Schleifeninvarianten an:

(a) **while** $\neg(X = 1)$ **do** $(Y := Y * X; X := X - 1)$ Fakultät, vgl. Vorlesung / 2

(b) **while** $\neg(Y = 0)$ **do** $(Y := Y - 1; X := 2 * X)$ Potenzfunktion / 3