Modellierung und Programmierung 2

Hinweise Abgabe: http://moodle2.uni-leipzig.de/course/view.php?id=1206

Übungsserie 5

Formale Semantik und Verifikation

Aufgabe 17 (Denotationale Semantik) Definieren Sie für die folgenden Anweisungen die semantische Funktion F der denotationalen Semantik, in folgender Form:

$$F[\langle Anweisung \rangle](z) = \ldots$$

Die Menge der Zustände Z_p soll dabei als Menge von Abbildungen der Variablennamen V auf den Bereich der ganzen Zahlen $\mathbb{Z}$ aufgefasst werden. Verwenden die aus der Vorlesung und Übung bekannten Schreibweisen für Fallunterscheidungen und Variablenblegungen $z\,\langle v \leftarrow w \rangle$, welche für die Variablenbelegung z die Variable v auf den Wert w zuweist und alle anderen Variablen unverändert läßt.

(a) `(* leere Anweisung *)`

(b) `x := x + 3`

(c) `IF y = 0 THEN u := u + 1 ELSE v := v - 1 END`

Aufgabe 18 Gegeben sei das folgende MINI-Programm P:

```
1  read x, y;
2  y := 0;
3  while x ≠ 0 do
4        if y = 0 then
5           y := y + 1
6        else
7           y := y - 1
8        end;
9        x := x - 1
10 end;
11 write y.
```

(a) Geben Sie in der Schreibweise der operationalen Semantik an, wie P die Eingabe $x = 2$, $y = 42$ verarbeitet. Um Schreibarbeit zu sparen, vereinbaren wir die folgende Abkürzungen für die folgenden Programmteile von P:
$i \equiv$ `if y = 0 then y := y + 1 else y := y - 1 end`
$w \equiv$ `while x ≠ 0 do i; x := x - 1 end`

(b) Welche Funktion berechnet P?

(c) Erweitern Sie die operationale Semantik von MINI um die `repeat`-Anweisung. Damit entsteht die Sprache MINI+ mit folgender Syntax-Ergänzung:
Anweisung → ... | „repeat“ Anweisung „until“ Bezeichner „= 0“

Aufgabe 19 (Programmverifikation) Gegeben sei die folgende Spezifikation eines in einem Pseudocode-Programmteils, welche die Vorbedingung P_{SPEC} und eine Nachbedingung Q_{SPEC} angibt:

$$P_{SPEC} \equiv \{-42 \leq y \leq 42\}, Q_{SPEC} \equiv \{y \geq 0\}$$

Gegeben sei weiterhin folgendes Programmfragment:

```
1 x := -y;
2 IF x < 0 THEN y := y - x  ELSE y := √x END;
```

Genügt dieses Programmfragment dieser Spezifikation? Begründen Sie Ihre Aussage *formal*!

Aufgabe 20 (Programmverifikation) Im folgenden Pseudocode-Programmfragment sind die Vor- und Nachbedingungen gegeben, welche das Programm spezifizieren.

```
1 (* P_SPEC : (b ≥ −3) AND (b ≤ 3) *)
2 a := -b;
3 IF a ≥ 0 THEN b := a - b ELSE b := a + b END;
4 (* Q_SPEC : (b > 0) AND (b ≤ 4) *)
```

(a) Bestimmen Sie zur Nachbedingung Q_{SPEC} die schwächste Vorbedingung.

(b) Ist das Programmfragment korrekt? Begründen Sie Ihre Aussage *formal*!

Zusatzaufgabe 5 (Programmverifikation, Invariante) Gegeben ist das folgende Programmfragment. Seine Nachbedingung ist $a = n^3, n \in \mathbb{N}$. Es handelt sich also um einen etwas trickreichen Algorithmus, der die dritte Potenz einer natürlichen Zahl n ohne Multiplikation berechnet. ($6n$ kann einmal vorher bestimmt werden.)

```
1 a := 0;
2 b := 1;
3 c := 0;
4 while c < 6 * n do
5     a := a + b;
6     c := c + 6;
7     b := b + c;
8 end;
```

(a) Geben Sie die Invariante an und beweisen Sie diese durch Induktion.

(b) Beweisen Sie die Nachbedingung.

(c) Was fehlt noch für eine vollständige Verifikation?