

## 2. Übungsblatt

**Ausgabe:** 21.04.16**Abgabe:** 28.04.16

### 2.1 Operationale Semantik von Ausdrücken

3 Punkte

In der Vorlesung wurde die operationale Semantik von C0 eingeführt. Geben Sie die Ableitung der Auswertung folgender Ausdrücke in der operationalen Semantik unter einem gegebenen Zustand  $\sigma$  an:

- $3 * (5 + x)$  mit  $\sigma(x) = 6$
- $a \parallel (a \ \&\& \ b)$  mit  $\sigma(a) = 1$  und  $b$  undefiniert
- $(a \ \&\& \ b) \parallel a$  mit  $\sigma(a) = 1$  und  $b$  undefiniert

### 2.2 Maximale Korrektheit?

5 Punkte

Folgendes Stück Code soll in  $z$  das Maximum der beiden Variablen  $x$  und  $y$  berechnen:

```
{???}  
if (x >= y) z = x; else z = y;  
{???
```

Geben Sie eine geeignete Spezifikation an (ersetzen Sie ??? durch geeignete Vor- und Nachbedingungen), und zeigen Sie mit dem Hoare-Kalkül, dass das Program die Spezifikation erfüllt. Die Spezifikation sollte vollständig sein, d.h. andere Programme, die nicht das Maximum berechnen, ausschließen.

Geben Sie den Beweis durch vollständige Annotation des Programmes in der in Vorlesung und Übung eingeführten Notation an.

### 2.3 Quadratisch, praktisch, schnell

9 Punkte

Folgendes Stück C0 soll das Quadrat einer ganzen Zahl  $x$  berechnen.

Wir prüfen das erst einmal exemplarisch: geben Sie dazu die Auswertung des Programmes mittels operationaler Semantik für  $\sigma(x) = 2$  an. (Sie können hier die Auswertung der Ausdrücke abkürzen; geben Sie nur die Herleitung der Zustandsübergänge  $\rightarrow_{stmt}$  an.)

```
{x = X}  
p = 0;  
while (x != 0) {  
  x = x - 1;  
  p = p + 2 * x + 1;  
}  
{p = X * X}
```

Wenn die Auswertung den gewünschten Wert ergibt, beweisen Sie dann mit dem Hoare-Kalkül in der Notation der vorangegangenen Aufgabe die partielle Korrektheit des Programmes.

## 2.4 Bedingungen

3 Punkte

Wir betrachten die Syntax für C0 aus der Vorlesung:

**Aexp**  $a ::= \mathbf{N} \mid \mathbf{Loc} \mid a_1 + a_2 \mid a_1 - a_2 \mid a_1 * a_2 \mid a_1 / a_2$

**Bexp**  $b ::= \mathbf{0} \mid \mathbf{1} \mid a_1 == a_2 \mid a_1 != a_2 \mid a_1 <= a_2 \mid !b \mid b_1 \&\&b_2 \mid b_1 || b_2$

**Exp**  $e ::= \mathbf{Aexp} \mid \mathbf{Bexp}$

**Stmt**  $c ::= \mathbf{Loc} = \mathbf{Exp}; \mid \text{if}(b) \ c_1 \ \text{else} \ c_2 \mid \text{while}(b) \ c \mid \{c^*\}$

In C0 gibt es auch den ternären Bedingungsoperator  $B ? E_1 : E_2$ , sowohl für arithmetische als auch boolesche Ausdrücke.

- Erweitern Sie die Syntax für arithmetische Ausdrücke um den ternären Bedingungsoperator.
- Definieren Sie entsprechend die operationalen Semantik für den Bedingungsoperator durch Angabe der geeigneten Regeln. Achten Sie auf Vollständigkeit und Striktheit — wann ist der gesamte Ausdruck undefiniert?
- Erweitern Sie die Syntax auch für boolesche Ausdrücke um den ternären Bedingungsoperator und definieren Sie die operationale Semantik entsprechend.