Formale Grundlagen der Informatik 3

Prof. Dr. Christoph Walther / Visar Januzaj, Nathan Wasser
Technische Universität Darmstadt — Wintersemester 2011/12

Hausübung 2

Geben Sie die handschriftliche Lösung für diese Übung **am 20. Dezember 2011 vor oder nach der Vorlesung** von 16:15 bis 17:55 in S2|02/C205 ab. Heften Sie Ihre Blätter **oben links** zusammen und versehen Sie alle Blätter **gut lesbar** mit Ihren Namen und Ihren Matrikel-Nummern sowie dem Namen Ihres Tutors. Es können **maximal vier** Übungsteilnehmer eine gemeinsam erarbeitete Lösung einreichen.

Lesen Sie vor dem Anfertigen einer Lösung die Aufgaben gründlich durch. Verwenden Sie Definitionen und Notationen wie im Rahmen dieser Veranstaltung vorgestellt. Für Fragen bezüglich dieser Übung können Sie das Forum zur Veranstaltung sowie die Sprechstunden der Tutoren nutzen.

Der Fachbereich Informatik misst der Einhaltung der Grundregeln der wissenschaftlichen Ethik großen Wert bei. Zu diesen gehört auch die strikte Verfolgung von Plagiarismus. Mit der Abgabe einer Lösung bestätigen Sie, dass Sie die alleinigem Autoren des gesamten Materials sind. Bei Unklarheiten zu diesem Thema finden Sie weiterführende Informationen unter `http://www.informatik.tu-darmstadt.de/Plagiarism` oder sprechen Sie Ihren Betreuer an.

Aufgabe 2.1 (Matching) (15 Punkte)

Bestimmen Sie den jeweils minimalen Matcher für die folgenden Matchingprobleme, falls ein Matcher existiert. Geben Sie dazu jeweils eine Herleitung im Matchingkalkül an, aus der dieser Matcher hervorgeht. Existiert kein Matcher, dann geben Sie alle erfolglosen Herleitungen an. Geben Sie in jedem Schritt die verwendete Regel an.

Beachten Sie: $x, y, z \in \mathcal{V}$ und $a, b, f, g, h \notin \mathcal{V}$

(a) Pattern: $t_a = f(f(g(x, x)))$, Target: $q_a = f(f(g(f(a), a)))$

(b) Pattern: $t_b = h(g(b, a), x, y)$, Target: $q_b = h(g(b, a), f(b), f(b))$

(c) Pattern: $t_c = g(f(z, b))$, Target: $q_c = g(f(b, a))$

Aufgabe 2.2 (Berechnungskalkül) (19 Punkte)

Betrachten Sie das Programm P mit folgender Prozedurdefinition:

```
function h2.2(x : ℕ, y : ℕ) : ℕ <=
if ?0(x)
  then h2.2(+(x), y)
  else if ?0(y)
         then x
         else h2.2(-(x), -(y))
       end_if
end_if
```

(a) (14,5 Punkte) Bestimmen Sie $eval_P(\texttt{h2.2}(\texttt{3}, \texttt{2}))$, indem Sie eine geeignete Herleitung im Berechnungskalkül angeben. Geben Sie dabei in jedem Schritt die verwendete Regel an.
Hinweis: Verwenden Sie die Kurzschreibweise für Konstruktorgrundterme vom Typ $\mathbb{N}$, z.B. `2` statt `succ(succ(0))` und `3` statt `succ(succ(succ(0)))`.

(b) (4,5 Punkte) Zeigen Sie, dass die durch `h2.2` berechnete Funktion nicht idempotent ist, dass also $\forall x : \mathbb{N}.\ h2.2(x, x) = x$ nicht gilt, indem Sie eine geeignete Substitution σ angeben und mit dem Berechnungskalkül nachweisen, dass $eval_P(\sigma(h2.2(x, x))) \neq eval_P(\sigma(x))$. Geben Sie dabei in jedem Schritt die verwendete Regel an.

Aufgabe 2.3 (Exception Guard) (6 Punkte)

Geben Sie für die Prozeduren `h2.3a`, `h2.3b` und `h2.3c` jeweils die Exception Guard $except_{h2.3a}[\mathtt{x}, \mathtt{y}]$, $except_{h2.3b}[\mathtt{n}]$ und $except_{h2.3c}[\mathtt{n}]$ an.

```
function h2.3a(x : ℕ, y: ℕ) : bool <=
if ?0(x)
  then ⋆
  else if x > y
         then if ?0(y)
                then ⋆
                else if ?0(pred(y))
                       then h2.3a(x, y)
                       else ⋆
                     end_if
              end_if
         else if ?0(pred(y))
                then h2.3a(y, x)
                else ⋆
              end_if
       end_if
end_if

function h2.3b(n : ℕ) : ℕ <= ⋆

function h2.3c(n : ℕ) : ℕ <= h2.3b(n)
```