Technische Universität München
Fakultät für Informatik
Prof. Dr. M. Broy

SS 2003
Übungsblatt 6
23. Mai 2003

Übungen zur Vorlesung Einführung in die Informatik IV

Aufgabe 23 Primitiv rekursive Funktionen

Geben Sie eine primitiv rekursive Definition für die folgende Funktion an:

```
fct sq = (nat n) nat:
  ⌈ var nat s := 0;
    for i := 0 to n - 1 do s := s + 2 * i + 1 od;
    s ⌋
```

Aufgabe 24 μ-rekursive Funktionen

Geben Sie eine μ-rekursive Definition der folgenden partiellen Funktion `psqrt` an, die die Quadratwurzel natürlicher Zahlen berechnet:

```
fct psqrt = (nat n) nat:
  ⌈ var nat y := 0;
    while n ≠ y * y do y := y + 1 od;
    y ⌋
```

Verwenden Sie hierbei die μ-rekursive Definition der partiellen Subtraktion!

Aufgabe 25 Äquivalenz von μ-Rekursion und while-Programmen auf Registermaschinen

(a) Gegeben sei die folgende μ-rekursive Funktion zur Berechnung der partiellen Subtraktion zweier natürlicher Zahlen

$$a \dot{-} b = \mu(h)(a,b),$$

wobei die primitiv rekursiv definierte Funktion $h : \mathbf{N}^3 -> \mathbf{N}$ definiert ist durch die Gleichung

$$h(a,b,y) = \mathrm{sub}(b+y,a) + \mathrm{sub}(a,b+y)$$

(sub und + seien die primitiv rekursiv definierte Funktionen für die totale Subtraktion bzw. Addition von natürlichen Zahlen). In dieser Aufgabe soll ein Registermaschinen-Programm angegeben werden, das $a \dot{-} b$ berechnet.

(i) Setzen Sie die Berechnung von $h(a,b,y)$ in ein Registermaschinen-Programm um. falls die Werte von a, b, und y in den Registern $s_{a,\ b}$ bzw. s_y gespeichert sind. Nach Beendigung der Berechnung sollen wieder die Werte in den Registern $s_{a,\ b}$ und s_y wiederhergestellt sein.

(ii) Setzen Sie die μ-Rekursion in ein Registermaschinen-Programm um.

(b) Zeigen Sie: Jedes Registermaschinen-Programm ist in eine μ rekursive Funktion umsetzbar, indem Sie für alle Programm-Konstrukte der Registermaschine Umsetzungen in ein μ-rekursives Programm angeben.

Aufgabe 26 (H) Äquivalenz von primitiver Rekursion und For-Programmen auf Registermaschinen

Durch

$$\texttt{for}_i(M) =_{def} \texttt{while}_i(M;\ pred_i)$$

sei innerhalb eines RM-Programmes eine for-Schleife gegeben, wobei das Programm das Register i nicht verändert. Wir beschränken uns in dieser Aufgabe auf solche RM-Programme, die nur über die Grundfunktionen ε, $\texttt{succ}_i$, $\texttt{pred}_i$, die Komposition $M_1;\ M_2$ und die so definierten for-Schleifen verfügen. Funktionen, die durch derartige Programme berechenbar sind, nennen wir for-berechenbar.

(a) Zeigen Sie: Eine Funktion ist primitiv rekursiv genau dann, wenn sie for-berechenbar ist.

(b) Diskutieren Sie Unterschiede der durch primitive Rekursion und durch μ-Rekursion definierbaren Funktionen.

Aufgabe 27 (P) Rekursionsoperator

In dieser Aufgabe soll das Rekursionsschema für primitiv rekursive Funktionen an Hand eines Anwendungsbeispiels umgesetzt werden. Hierfür wird der Rekursionsoperator *pr* für einstellige natürlichzahlige Funktionen $f : \mathbb{N} \to \mathbb{N}$, in Gofer definiert durch

```
pr :: Int -> ((Int, Int) -> Int) -> (Int -> Int)
pr g h = f where
                 f(0)   = g
                 f(n+1) = h(n, f(n))
```

zur Berechnung der Fakultätsfunktion herangezogen.

(a) Geben Sie eine primitiv rekursive Definition für die Fakultätsfunktion $fac : \mathbb{N} \to \mathbb{N}$ an.

(b) Implementieren Sie in Gofer die erforderlichen primitiv rekursiven Grundfunktionen (Nullfunktion, Nachfolgerfunktion, Projektionsfunktionen) sowie geeignete Kompositionsoperatoren für ein- und zweistellige natürlichzahlige Funktionen. Die Definition der zusätzlich benötigten Multiplikation kann vereinfacht als `mult(x, y) = x * y` angenommen werden (`mult` ist offensichtlich ebenfalls eine primitiv rekursive Funktion).

(c) Setzen Sie das in Teilaufgabe a) entwickelte Rekursionschema unter Verwendung der in b) implementierten Konstrukte zur Berechnung der Fakultät um. Testen Sie die korrekte Funktionalität der Umsetzung durch ausgewählte Berechnungen.