3. Übungsblatt zu Programmierung 1, WS 2012/13

Prof. Dr. Gert Smolka, Sigurd Schneider, B.Sc.

www.ps.uni-sb.de/courses/prog-ws12/

Lesen Sie im Buch: Kapitel 3

Aufgabe 3.9 Deklarieren Sie eine Prozedur $prod : (int \to int) \to int \to int$, die für $n \geq 0$ die Gleichung

$$prod\, f\, n = 1 \cdot (f\, 1) \ \dots\ \cdot (f\, n)$$

erfüllt. Deklarieren Sie außerdem mithilfe von *prod* eine Prozedur $fac : int \to int$, die für $n \geq 0$ die Fakultät $n!$ berechnet (siehe Aufgabe 1.26 auf S. 21). Die Prozedur *fac* soll nicht rekursiv sein.

Aufgabe 3.10 Deklarieren Sie mithilfe der höherstufigen Prozedur *sum* eine Prozedur $sum' : (int \to int) \to int \to int \to int$, die für $k \geq 0$ die Gleichung

$$sum'\, f\, m\, k = 0 + f(m+1) \ \cdots + f(m+k)$$

erfüllt. Die Prozedur sum' soll nicht rekursiv sein.

Aufgabe 3.11 Geben Sie die Baumdarstellung des folgenden Typs an:

$$(int \to bool) \to (bool \to real) \to int \to real$$

Aufgabe 3.12 Geben Sie geschlossene Abstraktionen an, die die folgenden Typen haben:

a) $(int * int \to bool) \to int \to bool$
b) $(int \to real) \to (bool \to int) \to int * bool \to real * int$

Die Abstraktionen sollen nur mit Prozeduranwendungen, Tupeln und Bezeichnern gebildet werden. Konstanten und Operatoren sollen nicht verwendet werden.

Aufgabe 3.14 Deklarieren Sie eine Prozedur $mul : int \to int \to int$, die das Produkt zweier Zahlen x und $n \geq 0$ gemäß der Gleichung

$$x \cdot n = 0 \underbrace{+ x \ \dots\ + x}_{n\text{-mal}}$$

durch Addieren berechnet. Die Prozedur *mul* soll mithilfe der Prozedur *iter* formuliert werden und nicht rekursiv sein.

Aufgabe 3.15 Geben Sie eine Abstraktion e an, sodass die Ausführung des Ausdrucks $first\ x\ e$ für alle x divergiert.

Aufgabe 3.16 Deklarieren Sie mit *first* eine Prozedur $divi : int \to int \to int$, die zu $x \geq 0$ und $m > 0$ dasselbe Ergebnis liefert wie Division mit dem Operator *div*. Hinweis: Für $x \geq 0$ und $m > 0$ liefert *div* die größte ganze Zahl k mit $k \cdot m \leq x$.

Aufgabe 3.33 (Exists) Deklarieren Sie eine Prozedur

$$exists : int \to int \to (int \to bool) \to bool$$

die testet, ob eine Prozedur für mindestens eine Zahl zwischen zwei gegebenen Zahlen den Wert *true* liefert. Orientieren Sie sich an der Prozedur *forall*.

Aufgabe 3.36 Schreiben Sie eine Prozedur $reduce : int \to int \to int * int$, die zu zwei Zahlen $n, p \geq 2$ das eindeutig bestimmte Paar (m, k) liefert, sodass $n = m \cdot p^k$ gilt und m nicht durch p teilbar ist.

Aufgabe 3.37 Deklarieren Sie mit *iterup* eine Prozedur

a) *power*, die zu x und n die Potenz x^n liefert.
b) *fac*, die zu $n \geq 0$ die n-te Fakultät $n!$ liefert.
c) *sum*, die zu f und n die Summe $0 + f\,1 \ldots + f\,n$ liefert.
d) *iter'*, die zu n, s und f dasselbe Ergebnis liefert wie $iter\ n\ s\ f$.

Aufgabe 3.27 Geben Sie Deklarationen an, die monomorph getypte Bezeichner wie folgt deklarieren:

$$a : int * unit * bool$$
$$b : unit * (int * unit) * (real * unit)$$
$$c : int \to int$$
$$d : int * bool \to int$$
$$e : int \to real$$
$$f : int \to real \to real$$
$$g : (int \to int) \to bool$$

Verzichten Sie dabei auf explizite Typangaben und verwenden Sie keine Operator- und Prozeduranwendungen.