DEPENDABLE SYSTEMS AND SOFTWARE

Fachrichtung 6.2 — Informatik
Prof. Dr.-Ing. Holger Hermanns
Dipl.-Inform. Lijun Zhang

Übungsblatt 10 (Programmierung I)

Lesen Sie im Skript: Kapitel 8 und Kapitel 9

Aufgabe 10.1: (Kleinstes Element, minimales Element und untere Schranke)
Geben Sie präzise Definitionen für die Begriffe kleinstes Element, minimales Element und untere Schranke an.

Aufgabe 10.2: (Ordnungen in Graphsicht)
Sei R eine Ordnung. Charakterisieren Sie die folgenden Begriffe aus der Graphsicht.

(a) $x \in \mathit{Ver}\ R$ ist minimales Element von $\mathit{Ver}\ R$.

(b) $x \in \mathit{Ver}\ R$ ist kleinstes Element von $\mathit{Ver}\ R$.

(c) $x \in X$ ist kleinstes Element von $X \subseteq \mathit{Ver}\ R$

(d) $x \in \mathit{Ver}\ R$ ist untere Schranke von $X \subseteq \mathit{Ver}\ R$.

Aufgabe 10.3: (Ordnungen)
Geben Sie eine lineare Ordnung R und eine nichtleere Menge $X \subseteq \mathit{Ver}\ R$ an, sodass X eine untere und eine obere Schranke hat, aber weder minimale noch maximale Elemente.

Aufgabe 10.4: (Hasse-Diagramme)
Zeichnen Sie die Hasse-Diagramme der folgenden Ordnungen:

(a) $NO(\{0, 2, 3, -3, -2\})$.

(b) $IO(\{2, 4\})$.

(c) $SO(K)$, wobei K die Menge aller Konsitutenten von $\{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$ ist.

Aufgabe 10.5: (Nicht-lineare Ordnungen)
In § 5.2 haben wir gesehen, wie lineare Ordnungen in Standard ML durch Prozeduren dargestellt werden können. Mit einer zusätzlichen Orientierung U (undefined) können wir auch nicht-lineare Ordnungen darstellen:

```
datatype orientation = L | E | G | U
type 'a ord = 'a * 'a -> orientation
```

Die eine Ordnung darstellende Prozedur liefert für ein Paar (x, y) genau dann die Orientierung U, wenn weder (x, y) noch (y, x) in der Ordnung ist. Wenn (x, y) in der Ordnung ist, wird für $x = y$ die Orientierung E (equal) geliefert, und sonst L (less). Wenn keiner der bisherigen Fälle zutrifft, wird G (greater) geliefert.

(a) Schreiben Sie eine Prozedur $\mathit{convert} : (\alpha * \alpha \to \mathit{order}) \to \alpha\ \mathit{ord}$, die die Darstellung einer linearen Ordnung in die Darstellung einer allgemeinen Ordnung konvertiert.

(b) Schreiben Sie eine Prozedur $\mathit{invert} : \alpha\ \mathit{ord} \to \alpha\ \mathit{ord}$, die zu einer Ordnung die inverse Ordnung liefert.

(c) Schreiben Sie eine Prozedur $\mathit{prod} : \alpha\ \mathit{ord} \to \beta\ \mathit{ord} \to (\alpha * \beta)\ \mathit{ord}$, die das lexikalische Produkt zweier Ordnungen liefert.

(d) Schreiben Sie eine Prozedur $\mathit{lex} : \alpha\ \mathit{ord} \to \alpha\ \mathit{list}\ \mathit{ord}$, die zu einer Ordnung die ihr entsprechende lexikalische Ordnung für Listen liefert.

Aufgabe 10.6: (Induktion)
Beweisen Sie mit Induktion: $\forall n \in \mathbb{N}\ \exists k \in \mathbb{N} : n^3 - n = 3 \cdot k$. Geben Sie die Ihrem Beweis zugrunde liegende Grundmenge, Aussagemenge, Induktionsrelation und Induktionsannahme an.

Aufgabe 10.7: (Multiplikation)
Konstruieren Sie eine terminierende Prozedur $p : \mathbb{Z} \times \mathbb{N} \to \mathbb{Z}$, die die Funktion $f = \lambda\ (x, n) \in \mathbb{Z} \times \mathbb{N}.\ xn$ ohne Multiplikation mit Addition berechnet.

(a) Zeigen Sie die Terminierung Ihrer Prozedur p mit einer natürlichen Terminierungsfunktion.

(b) Argumentieren Sie auf der Grundlage von Proposition 9.6, warum Ihre Prozedur p für f korrekt ist.

(c) Erweitern Sie Ihre Prozedur p zu einer terminierenden Prozedur $p' : \mathbb{Z}^2 \to \mathbb{Z}$.

(d) Zeigen Sie die Terminierung Ihrer Prozedur p' mit einer natürlichen Terminierungsfunktion.

(e) Argumentieren Sie auf der Grundlage von Proposition 9.7, warum Ihre Prozedur p' für f korrekt ist.

Aufgabe 10.8: (Korrektheitsbeweis)
Zeigen Sie, dass die Prozedur

$p : \mathbb{N} \to \mathbb{N}$
$p\ n =$ if $n < 1$ then 1 else $p(n-1) + 2n + 1$

die Funktion $\lambda\ n \in \mathbb{N}.\ (n+1)^2$ berechnet.

Aufgabe 10.9: (Korrektheitsbeweise)
Zeigen Sie, dass die Prozedur

$euclid : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$
$euclid(x, y) =$ if $y = 0$ then x else $euclid(y, x \bmod y)$

die Funktion $\lambda(x, y) \in \mathbb{N}^2.$ if $y = 0$ then x else $ggt(x, y)$ berechnet. (Definition *ggt* siehe Script).

Aufgabe 10.10: (Summe ungerader Zahlen)
Geben Sie eine rekursive Prozedur $p : \mathbb{N}^+ \to \mathbb{N}$ an, die für $n \in \mathbb{N}^+$ die Summe $1 + 3 + \cdots + (2n-1)$ der ungeraden Zahlen von 1 bis $2n-1$ berechnet. Beweisen Sie, dass Ihre Prozedur die Funktion $\lambda\ n \in \mathbb{N}^+.\ n^2$ berechnet.

Aufgabe 10.11: (Summe der Quadratzahlen)
Geben Sie eine rekursive Prozedur $p : \mathbb{N} \to \mathbb{N}$ an, die für $n \in \mathbb{N}$ die Summe $0^2 + 1^2 + \cdots + n^2$ berechnet. Beweisen Sie, dass Ihre Prozedur für alle $n \in \mathbb{N}$ das Ergebnis $\frac{n}{6}(2n^2 + 3n + 1)$ liefert.

Aufgabe 10.12: (Potenzen)
Die Potenzen x^n lassen sich mit einer endrekursiven Prozedur bestimmen, die die folgende Funktion berechnet:

$poweri : \mathbb{Z} \to \mathbb{Z} \to \mathbb{N} \to \mathbb{Z}$
$poweri\ a\ x\ n = a \cdot x^n$

(a) Konstruieren Sie eine endrekursive Prozedur *poweri*, die die Funktion *poweri* berechnet und beweisen Sie die Korrektheit Ihrer Prozedur (Korrektheit per Konstruktion).

(b) Schreiben Sie in Standard ML eine Prozedur *power*, die Potenzen mithilfe einer endrekursiven Prozedur *poweri* berechnet.

Aufgabe 10.13: (Korrektheitsbeweise)
Die Quersumme einer Zahl können wir wie folgt definieren:

$cross : \mathbb{N} \to \mathbb{N}$
$cross\ n = n$ für $n < 10$
$cross\ n = n \bmod 10 + cross(n \operatorname{div} 10)$ für $n \geq 10$

Beachten Sie, dass $n \bmod 10$ die letze Ziffer der Zahl n liefert.

(a) Konstruieren Sie eine endrekursive Prozedur *crossi*, die die Funktion
$f : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$
$f\ a\ n\ = a + cross\ n$
berechnet und beweisen Sie die Korrektheit Ihrer Prozedur (Korrektheit per Konstruktion).

(b) Schreiben Sie in Standart ML eine Prozedur *cross*, die die Quersumme einer Zahl mithilfe der endrekursiven Prozedur *crossi* berechnet.