7. Übungsblatt zu Programmierung (WS 1999/2000)

Andreas Rossberg, Prof. Gert Smolka
Abgabe: 10. Dezember 1999 vor der Vorlesung

Hinweis: Die Aufgaben 7.4 bis 7.5 brauchen nur schriftlich abgegeben zu werden.

Aufgabe 7.1: Spielkarten (10 Punkte)

Nehmen Sie an, Sie möchten die 32 verschiedenen Karten eines Skatspiels in SML repräsentieren. Eine Karte kann durch zwei Eigenschaften beschrieben werden:

- ihre Farbe (Kreuz ♣, Pik ♠, Herz ♡, Karo ♢)
- ihren Rang (eine Zahl von 7 bis 10, eines der Bilder Bube, Dame, König, oder ein As)

Abhängig von Rang und Farbe hat jede Karte einen relativen Wert:

- Karten der Farbe Kreuz sind mehr wert als Karten der Farbe Pik, Pik-Karten mehr als Herz-Karten, Herz ist höher als Karo.
- Innerhalb einer Farbe ist das As die höchste Karte, gefolgt von König, Dame, Bube und den Zahlkarten in absteigender Reihenfolge.

(a) Geben Sie geeignete Deklarationen für zwei Konstruktortypen `farbe` und `rang` an, um die Eigenschaften von Karten zu beschreiben, so dass die Typdeklaration

```
type karte = farbe * rang
```

den Typ von Spielkarten definiert.

(b) Schreiben Sie eine Prozedur

```
compare: karte * karte -> order
```

die zwei Karten entsprechend ihres relativen Wertes vergleicht und eine entsprechenden Ordnungswert liefert. Tipp: Am einfachsten ist es, zunächst Hilfsprozeduren zu schreiben, die jeder Karte eine eindeutige Zahl zuordnen, und dann diese Zahlen mit `Int.compare` zu vergleichen.

(c) Schreiben Sie nun eine Prozedur

```
isSkatBlatt: karte list -> bool
```

die feststellt, ob eine Liste von Karten gerade ein vollständiges Skatblatt (32 Karten) ist. Sie können dazu die Prozedur `ListSort.sort` von Moscow ML benutzen.

Aufgabe 7.2: Symbolisches Differenzieren (12 Punkte)

Das Differenzieren von mathematischen Funktionen geschieht nach einfachen Regeln, es lässt sich deshalb leicht automatisieren. Im folgenden sollen Sie eine Prozedur entwickeln, die die symbolische Ableitung einfacher Funktionen über einer Unbekannten $x \in \mathbb{R}$ berechnen kann, zum Beispiel $(x^3 + \frac{4}{5}x^2 + x + 2)' = 3x^2 + \frac{8}{5}x + 1$.

Die zu verarbeitenden Ausdrücke sollen wie folgt repräsentiert werden:

```
datatype expr = Con of int            (* c *)
              | X                     (* x *)
              | Add of expr * expr    (* u+v *)
              | Mul of expr * expr    (* u·v *)
              | Pow of expr * int     (* u^n *)
```

(a) Geben Sie einen Ausdruck des Typs `expr` an, der das folgende Polynom $p(x)$ repräsentiert:

$$x^3 + \frac{4}{5}x^2 + x + 2$$

(b) Schreiben Sie eine Prozedur `derive: expr -> expr`, die einen beliebigen Ausdruck nach x differenziert. Verwenden Sie dazu folgende Regeln:

$$\begin{aligned} c' &= 0 \\ x' &= 1 \\ (u+v)' &= u' + v' \\ (u \cdot v)' &= u' \cdot v + u \cdot v' \\ (u^n)' &= n \cdot u^{n-1} \cdot u' \end{aligned}$$

Der resultierende Ausdruck darf redundante Teilausdrücke (wie $x + 0$) enthalten.

(c) Die Ergebnisse, die Ihre Prozedur `derive` liefert, werden im allgemeinen viele unnötige Teilausdrücke enthalten. Schreiben Sie aus diesem Grund zunächst eine Prozedur `simplify1: expr -> expr`, die eine Formel vereinfacht, indem sie eine der folgenden Gleichungen versucht anzuwenden:

$$\begin{aligned} 0 + u &= u & \qquad u + 0 &= u \\ 0 \cdot u &= 0 & \qquad u \cdot 0 &= 0 \\ 1 \cdot u &= u & \qquad u \cdot 1 &= u \\ u^0 &= 1 & \qquad u^1 &= u \end{aligned}$$

Vereinfacht werden soll nur auf oberster Ebene, Unterausdrücke sollen also unverändert bleiben. Falls keine Vereinfachung möglich ist, bleibt der ganze Ausdruck unverändert.

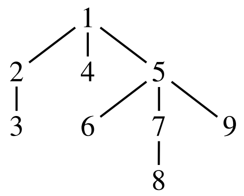
(d) Schreiben Sie jetzt eine Prozedur `simplify: expr -> expr`, die einen Ausdruck vollständig vereinfacht, indem zunächst alle Unterausdrücke vereinfacht werden und dann `simplify1` angewendet wird. Der Ergebnisausdruck sollte dann keine Teilausdrücke mehr enthalten, die sich noch nach obigen Regeln vereinfachen lassen.

Aufgabe 7.3: Terme (9 Punkte)

Gegeben sei folgende Deklaration für Terme über beliebigen Typen:

```
datatype 'a term = T of 'a * 'a term list
```

(a) Schreiben Sie eine Deklaration, die den Bezeichner `t` an einen Wert bindet, der den folgenden Baum repräsentiert:



(b) Schreiben Sie eine Prozedur `size: 'a term -> int`, die die Größe eines Terms berechnet. Tipp: Benutzen Sie `map` und `foldl`.

(c) Schreiben Sie eine Prozedur `node: int list * 'a term -> 'a`, die die Marke eines Knotens liefert. So soll beispielsweise für `t` aus (a) gelten:

```
node([3,2], t) = 7
```

(d) Schreiben Sie eine Prozedur `mapTerm: ('a -> 'b) -> 'a term -> 'b term`, die eine Prozedur auf jede Marke eines Terms anwendet (analog zur Prozedur `map` für Listen).

Aufgabe 7.4: Teilbäume (4 Punkte)

Gegeben sei der geordnete Baum

[]	4
[1]	3
[2]	0
[2,1]	4
[2,2]	2
[2,2,1]	1
[2,2,2]	6
[2,3]	7
[3]	2
[3,1]	4
[3,1,1]	0
[3,2]	5

Geben Sie die folgenden geordneten Teilbäume in Tabellenform an:

(a) den von [3] aus erreichbaren Teilbaum

(b) den von [2,2] aus erreichbaren Teilbaum

Aufgabe 7.5: Balancierte Binärbäume (8 Punkte)

(a) Zeigen Sie, dass für alle Binärbäume B (z.B. in Graphrepräsentation) mit b Blättern und Tiefe t gilt:

$$b \leq 2^t$$

(b) Sei B ein Binärbaum mit b Blättern und Tiefe t für den gilt:

$$b = 2^t$$

Beweisen Sie, dass B balanciert ist.

Aufgabe 7.6: Weihnachtsfeier (10 000 000 Quasipunkte)

Es sei der 9. Dezember, 16h. Betrachten Sie sich als eingeladen zur Weihnachtsfeier der Fachschaft. Begeben Sie sich in den Seminarraum 015, um den dortigen Festivitäten beizuwohnen.

(a) Erkennen Sie einzelne Kommilitonen wieder?

(b) Gibt es Glühwein? Falls ja, wie schmeckt er?

(c) Erkennen Sie zwei Stunden später ihre Kommilitonen aus (a) immer noch?