Ludwig-Maximilians-Universität München
Institut für Informatik
Prof. Dr. M. Wirsing, M. Hammer

SoSe 2009
Übungsblatt 7

Programmierung und Modellierung

Aufgabe 7-1 Pattern Matching

Übersetzen Sie die folgenden SML-Funktionen, so dass diese Pattern Matching verwenden. Ein `if` soll in Ihrer Lösung nicht mehr vorkommen. Geben Sie Ihre Lösung in einer Datei `7-1.sml` ab.

a)
```
fun intToBit n = if n = 0 then 0 else 1;
```

b)
```
fun konstant(l) = if l = nil
                  then true
                  else if tl(l) = nil
                       then true
                       else hd(l) = hd(tl(l)) andalso konstant(tl(l));
```

c) Unter Verwendung der Datei `6-2.sml` vom letzten Übungsblatt:

```
fun linkerPfad b = if (isempty b)
                   then []
                   else (root b) :: linkerPfad (left b);
```

Aufgabe 7-2 Quicksort

Quicksort (Hoare, 1962) ist ein bekanntes Sortierverfahren, das auf dem *Divide-and-Conquer*-Prinzip basiert. Für eine nicht-leere Liste wird ein beliebiges Element herausgenommen – das sogenannte *Pivot*-Element. Die Liste wird dann aufgeteilt in Elemente, die kleiner als das Pivot-Element sind, und in Element, die größer (oder gleich) dem Pivot-Element sind. Diese beiden Listen werden rekursiv mit Quicksort sortiert, dann werden sie zusammengehängt, mit dem Pivot-Element in der Mitte.

Laden Sie von der Vorlesungswebsite die Datei `7-2.sml` herunter, und ergänzen Sie Ihre Implementierungen. Wenn Sie wollen, können Sie einige Tests angeben; ein paar dafür nützliche Funktionen sind in der Datei `7-2.sml` definiert.

a) Implementieren Sie eine höhere Funktion

   `split: ('a -> bool) -> 'a list -> 'a list * 'a list`,

   die für ein Prädikat und eine Liste ein Paar von Listen berechnet, wobei in der ersten Liste alle die Elemente der Eingabe sind, die das Prädikat erfüllen, und in der zweiten alle die, die es nicht erfüllen.

b) Verwenden Sie Ihre Funktion `split`, um eine Funktion `quicksort` zu erstellen, die den Quicksort-Algorithmus implementiert.

c) Schreiben Sie eine SML-Funktion `quicktree`, die – analog zur Funktion `quicksort` – aus einer unsortierten Liste einen binären Suchbaum erstellt. Das Pivot-Element wird zur Wurzel, die Teilbäume werden jeweils rekursiv erstellt. Ist der resultierende Baum ausgeglichen?

Hinweis: Bei allen Teilaufgaben kann die Verwendung von `let P in Q end` hilfreich sein (siehe Foliensatz 5, Folie 14). In `P` können dabei lokale Deklarationen angegeben werden, auf die in `Q` zugegriffen werden kann. Sie können also z.B. in `P` mit `val (x,y) = ...` ein Paar definieren, dessen erstes Element an den Namen `x` und dessen zweites Element an den Namen `y` gebunden wird, und dann `x` und `y` in `Q` verwenden. Z.B. ergibt

```
let val (x,y) = (2,3) in x+y end;
```

den Wert 5.

Aufgabe 7-3 **Wiederholung: Typinferenz**

a) Geben Sie für die folgenden SML-Definitionen den prinzipalen Typ an. Natürlich sollen Sie dabei nicht den SML-Interpreter benutzen.

1. `fun apply f g = f(g);`
2. `fun pair x y = (x,y);`
3. `fun test x = apply (fn (x,y) => x) (pair 5 x);`

b) Leiten Sie die folgende Typaussagen ab:

1. {} ▶ `fn x => (fn y => x + y): int -> int -> int`
2. {} ▶ `fn x => (x + 5, real x) : int -> int * real`
   (mit {} ▶ `real: int -> real` gegeben)

Geben Sie Ihre Lösung als eine Datei `7-3.txt` ab. Beachten Sie dazu die Hinweise auf Übungsblatt 5.

Abgabe: Montag, den 22.6.2009, 12 Uhr, per UniWorx.