7. Aufgabenblatt zur Vorlesung

Informatik A

WiSe 2015/16

Wolfgang Mulzer

Abgabe am 8. Dezember 2015 bis 12 Uhr in die Tutorenfächer

Aufgabe 1 Uhrzeit *10 Punkte*

Wir führen den Datentypen `type Zeit = (Int,Int)` ein, der eine Uhrzeit als Paar (h, m) von Stunden- und Minutenwert darstellt ($h \in \{0, 1, \ldots, 23\}$, $m \in \{0, 1, \ldots, 59\}$).

Implementieren Sie die folgenden Funktionen. Geben Sie alle Signaturen an und machen Sie geeignete Testläufe. Versehen Sie Ihr Skript mit sinnvollen Kommentaren.

(a) Geben Sie Funktionen `addStd, addMin:: Zeit -> Int -> Zeit`, welche die Uhr um eine bestimmte Anzahl von Stunden bzw. Minuten vor oder zurück stellen.

Achtung: Beachten Sie mögliche Sprünge über Stunden- oder Tagesgrenzen. Zum Beispiel soll `addMin (23,58) 4` das Ergebnis `(0,2)` haben.

(b) Implementieren Sie eine Funktion `dauer:: Zeit -> Zeit -> Zeit`, die für zwei gegebene Zeiten `t1 t2` die Zeitdauer berechnet, wann zum ersten Mal nach der Zeit `t1` die Zeit `t2` angezeigt wird (bei Gleichheit ist das Ergebnis `(0,0)`).

(c) Implementieren Sie eine Funktion `anzeige:: Zeit -> String`, mit der eine Zeit (h, m) in einen String der Form "h:m Uhr" umgewandelt wird, wobei der Minutenwert immer zweistellig dargestellt werden soll.

Aufgabe 2 Cäsar-Chiffre *10 Punkte*

In der Vorlesung haben wir den Cäsar-Chiffre in Haskell gesehen.

(a) Verallgemeinern Sie das Skript aus der Vorlesung so, dass nun auch Großbuchstaben erlaubt sind.

(b) Verallgemeinern Sie das Skript aus der Vorlesung zu einem *Substitutionschiffre*. Implementieren Sie dazu Funktionen `kodiere, dekodiere:: [(Char, Char)] -> String -> String`, welche eine gegebene Zeichenkette kodieren bzw. dekodieren. Das erste Argument ist der *Schlüssel*. Er besteht aus einer Folge von Paaren von Zeichen. Dabei bedeutet ein Eintrag `(c,d)` im Schlüssel, dass das Zeichen `c` im Klartext durch das Zeichen `d` im verschlüsselten Text dargestellt wird. Die Zeichen, die nicht in der Schlüsselliste vorkommen, sollen nicht verändert werden. Sie können davon ausgehen, dass der Schlüssel wohldefiniert ist, so dass beim Dekodieren keine Mehrdeutigkeiten auftreten können.

Beispiele:

```
kodiere [('a','b'),('b','a'),('l','o'),('o','l')] ''hallo''
```

soll `''hbool''` liefern, und

```
dekodiere [('a','b'),('b','a'),('l', 'o'),('o', 'l')] ''hbool''
```

soll `''hallo''` ergeben.

(c) *(freiwillig, 5 Zusatzpunkte)* Implementieren Sie den Cäsar-Chiffre mit Hilfe des Substitionschiffre. Schreiben Sie dazu eine Funktion `erzeugeSchluessel:: Int -> [(Char, Char)]`, welche aus einem Schlüssel für den Cäsar-Chiffre einen Schlüssel für den Substitutionschiffre erzeugt.

Geben Sie alle Signaturen an und machen Sie geeignete Testläufe. Versehen Sie Ihr Skript mit sinnvollen Kommentaren.

Aufgabe 3 Funktionen höherer Ordnung *10 Punkte*

(a) Schreiben Sie eine Funktion `tausche`, welche als Eingabe eine zweistellige Funktion `f:: a -> b -> c` erhält und welche eine Funktion `f':: b -> a -> c` liefert, die das gleiche tut wie `f`, bei der aber die Reihenfolge der Argumente vertauscht ist.

(b) Schreiben Sie Funktionen `komplett`, `eins`, `behalteBis` und `verwirfBis`. Diese Funktionen sollen jeweils eine Liste `xs` vom Typ `a` und ein Prädikat `p:: a -> Bool` erhalten.

Die Funktion `komplett` soll genau dann `True` liefern, falls alle Elemente von `xs` das Prädikat `p` erfüllen. Die Funktion `eins` soll genau dann `True` liefern, falls mindestens ein Element von `xs` das Prädikat `p` erfüllt. Die Funktion `behalteBis` soll den längsten *Präfix* von `xs` liefern, in dem kein Element das Prädikat `p` erfüllt. Die Funktion `verwirfBis` soll den längsten Präfix von `xs` entfernen, in dem kein Element das Prädikat `p` erfüllt.

Ihre Funktionen sollen direkt rekursiv arbeiten.

(c) Definieren Sie `komplett` und `eins` aus (b) mit Hilfe von `map` und `foldl`.

Geben Sie alle Signaturen an und machen Sie geeignete Testläufe. Versehen Sie Ihr Skript mit sinnvollen Kommentaren.