

## Informatik A, WS 2016/17 — 8. Übungsblatt

Abgabe bis Freitag, 9. Dezember 2016, 12:00 Uhr, in die Fächer der Tutor/inn/en

Schreiben Sie von nun an *für jede Programmieraufgabe* eine Funktion `main :: IO ()`, die eine kleine Testsuite mit einer Handvoll Beispielen durchrechnet und die Ergebnisse ausgibt.

### 43. Inverse Funktionen, 10 Punkte

Schauen Sie die Definitionen der Funktionen `lines` und `unlines` (und der benötigten Hilfsfunktionen) in der Spezifikation des `HASKELL-Sprachstandards`<sup>1</sup> nach. Sind die Funktionen invers zueinander? Für welche  $x$  gilt

`lines (unlines x) == x?`

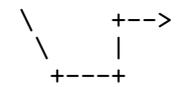
Für welche  $x$  gilt

`unlines (lines x) == x?`

Sie können QuickCheck zu Hilfe nehmen.

### 44. Spiegeln und Drehen von zweidimensionalen Mustern (10 Punkte)

In Aufgabe 41 haben wir Buchstabengrafiken erzeugt, die aus durch `'\n'` abgeschlossenen Zeilen bestehen und mit `putStr` ausgegeben werden. Zum Beispiel liefert `putStr "\ \ +-->\n \ \ | \n +---+\n"` das obige Bild. (Verkehrte Schrägstriche müssen in der Zeichenkette doppelt eingegeben werden, damit man sie von Sonderkodierungen wie `\n` unterscheiden kann.)



Schreiben Sie Funktionen `flipH`, `flipV`, und `flipD`, die eine solches Bild an einer horizontalen Achse, an einer vertikalen Achse, beziehungsweise an einer Diagonale von links oben nach rechts unten spiegeln. Schreiben Sie Funktionen `dreheL` und `dreheR`, die das Bild um  $90^\circ$  nach links beziehungsweise nach rechts drehen. Das Ergebnis soll bei dem Beispiel so aussehen:

<pre>flipH:  &gt;--+ \            \         +---+</pre>	<pre>flipV:  +----+         \              \  +--&gt;</pre>	<pre>flipD:  \         \          +         -         -         + +         -         -         + +         -         -         &gt;</pre>	<pre>dreheL:  &gt;         -         -         + +         -         -         +         \         \</pre>
---	---	--	--

Hinweis: Die Bibliotheksfunktion `reverse :: [a] -> [a]` kehrt die Reihenfolge einer Liste um. Manche Funktionen können aus anderen zusammengesetzt werden.

### 45. Faltung, 10 Punkte

- (a) Die Funktion `map` kann auch sehr knapp durch eine Faltung definiert werden:
 

```
map2 :: (a->b) -> [a] -> [b]
map2 h = foldr ((:).h) []
```

Geben Sie für *jeden* Teilausdruck in dieser Definition (also zum Beispiel auch für „`foldr`“ und „`.`“) den Datentyp in Abhängigkeit von `a` und `b` an.
- (b) Programmieraufgabe. Definieren Sie die Funktionen `filter`, `any`, `unlines` und `takeWhile` (siehe Fußnote <sup>1</sup>) als linke oder rechte Faltung. Verwenden Sie dabei neue Namen `filter'`, `any'`, usw.

### 46. Hornklauseln, 0 Punkte

Überprüfen Sie die folgende Boolesche Formel mit Resolution auf Erfüllbarkeit.

$$(\neg a \vee \neg b \vee \neg d) \wedge (\neg e) \wedge (\neg c \vee a) \wedge (c) \wedge (b) \wedge (\neg g \vee d) \wedge (g)$$

<sup>1</sup><https://www.haskell.org/onlinereport/haskell2010/haskellch9.html>