Fortgeschrittene funktionale Programmierung in Haskell

Übungszettel 1

Aufgabe 1.1:

Implementieren Sie eine einfache Verschiebechiffre[1], die jeden `Char` in einem `String` um eine gegebene Anzahl von Schritten verschiebt. Diese Funktion soll „in beide Richtungen“, also sowohl zum Ver- als auch zum Entschlüsseln nutzbar sein.

```
-- Für alle n soll gelten: (caesar (-n)) . (caesar n) = id
caesar :: Int -> String -> String
```

Um einen einzelnen `Char` zu „rotieren“ kann die Typklasseninstanz `Enum` von `Char` nützlich sein. Lesen Sie sich dafür die entsprechende Dokumentation in der *Prelude*[2] durch.

Aufgabe 1.2:

Implementieren Sie eine Funktion, die eine Datei (z.B. die mitgelieferte `gnu.txt`) von der Festplatte ausliest, alle Zeilen löscht, die nicht einem an Ihre Funktion übergebenen Prädikat (also einer Funktion `(String -> Bool)`) entsprechen und dann unter einem anderen Namen abspeichert.

```
filterFile :: (String -> Bool) -> FilePath -> IO ()
```

Die hierfür notwendigen `IO`-Funktionen finden Sie ebenfalls in der *Prelude*.

Aufgabe 1.3:

Hinweis: In dieser Aufgaben können an manchen Stellen *List Comprehensions* von Nutzen sein.

Aufgabenteil (a):

Schreiben Sie eine Funktion, die alle Primzahlen kleiner oder gleich einer Eingabezahl findet. Schreiben Sie außerdem eine Funktion, die eine endlose Liste von Primzahlen generiert (ohne gesonderte Eingabe). Hier sind zwei mögliche Typsignaturen:

```
finitePrimes   :: Integer -> [Integer]
infinitePrimes :: [Integer]
```

Aufgabenteil (b):

Implementieren Sie zwei Funktionen mit folgendem Verhalten. Sie können dafür Ihre Ergebnisse aus Aufgabenteil **(a)** verwenden.

- Gegeben einen Wert vom Typ `Integer`, geben Sie die Primfaktoren dieses Wertes als aufsteigende Liste zurück (Diese Liste ist eindeutig für jedes $n \in \mathbb{N}$)[3].

  ```
  primfaktoren :: Integer -> [Integer]
  ```

- Gegeben einen Wert vom Typ `Integer` soll ein Tupel von zwei Primzahlen ausgegeben werden, die summiert die eingegebene Zahl ergeben.

  ```
  goldbach :: Integer -> (Integer, Integer)
  ```

  (Das so ein Paar von Primzahlen für jedes $n \in \mathbb{N}$ existiert, ist die *Goldbachsche Vermutung*[4], eine bisher unbewiesenen Aussage der Mathematik)

[1] http://de.wikipedia.org/wiki/Caesar-Verschlüsselung
[2] http://hackage.haskell.org/package/base-4.8.0.0/docs/Prelude.html
[3] http://de.wikipedia.org/wiki/Primfaktorzerlegung#Fundamentalsatz_der_Arithmetik
[4] http://de.wikipedia.org/wiki/Goldbachsche_Vermutung