Praktische Informatik 3 WS 12/13

Christoph Lüth
Marcus Ermler
Christian Maeder
Martin Ring
Diedrich Wolter

1. Übungsblatt

Ausgabe: 24.10.12
Abgabe: 02.11.12

1.1 *Hexadezimales* *10 Punkte*

Implementieren sie zwei Funktionen

```
encode :: Integer→ String
decode :: String→ Integer
```

die eine ganze Zahl in *Hexadezimaldarstellung* ausgeben beziehungsweise einlesen. Die Ziffern von 10 bis 15 sollen wie üblich durch die Buchstaben `A` bis `F` dargestellt werden:

```
*Hex> encode 65347
"FF43"
*Hex> decode "FF43"
65347
```

Es ist hilfreich, dazu zwei Hilfsfunktionen

```
hexord :: Char→ Integer
hexchr :: Integer→ Char
```

zu definieren, welche eine einzelne Hexadezimalziffer ausgeben.

Formulieren Sie sinnvolle Eigenschaften der Funktionen encode und decode, die Sie in Ihren Testfällen benutzen.

1.2 *Beliebigemales* *10 Punkte*

Verallgemeinern Sie ihre Lösung der vorherigen Aufgabe zu einer Darstellung zu einer *beliebigen* Basis.

- Wie groß (und wie klein) kann die Basis sinnvollerweise werden?
- Welche Änderungen sind an den Signaturen der Funktionen nötig?
- Welche Eigenschaften der Funktionen gelten noch, welche müssen angepasst werden?

Hinweis: Folgende vordefinierte Funktionen können hilfreich sein:

— Konversionen zwischen Int und Integer:

```
toInteger :: Int→ Integer
fromInteger :: Integer→ Int
```

— Vordefinierte Funktionen auf Char und String

```
null :: String → Bool
head :: String → Char
tail :: String → String
init :: String → String
last :: String → Char
(:)  :: Char→ String→ String
(++) :: String → String
```

— Vordefinierte Funktionen auf Integer

```
div, mod  :: Integer→ Integer→ Integer
```

Die folgenden drei müssen mittels einer magischen import Inkantation am Anfang der Datei importiert werden:

```
import Data.Char
-- Konversionen zwischen Int und Char:
ord :: Char → Int
chr :: Int → Char
-- Konversionen zwischen Char:
toUpper :: Char→ Char
```

Ferner kann man sich folgende Hilfsfunktion definieren, die ein einzelnes Zeichen hinten an eine Zeichenkette anhängt (analog zu (:), auch cons genannt, daher der Name):

```
snoc :: String → Char → String
snoc s c = s ++ [c]
```

? *Verständnisfragen*

Auf allen Übungsblättern finden sich Verständnisfragen zur Vorlesung. Diese sind nicht Bestandteil der Abgabe, können aber im Fachgesprach thematisiert werden. Wenn Sie das Gefühl haben, diese Fragen nicht sicher beantworten zu können, wenden Sie sich gerne an Ihren Tutor, an Berthold Hoffmann in seiner Fragestunde, oder an den Dozenten.

1. Was bedeutet Striktheit, und welche in Haskell definierbaren Funktionen haben diese Eigenschaft?
2. Was ist ein algebraischer Datentyp, und was ist ein Konstruktor?
3. Was sind die drei Eigenschaften, welche die Konstruktoren eines algebraischen Datentyps auszeichnen, was ermöglichen sie und warum?