WiSe 2015/2016

Funktionale Programmierung

8. Übungsblatt (Abgabe: Mi., den 09. Dez. um 10:10 Uhr)

Prof. Dr. Margarita Esponda

Ziel: weitere Auseinandersetzung mit algebraischen Datentypen und Typ-Klassen. Auseinandersetzung mit abstrakten Datentypen.

1. Aufgabe (12 Punkte)

a) Definieren Sie einen abstrakten Datentyp **Queue**, in dem Sie eine Warteschlange mit den Operationen **enqueue** (Einfügen) , **dequeue** (Entfernen), **isEmpty** und **makeQueue** (Erzeugt eine leere Warteschlange) modellieren.

b) Definieren Sie eine geeignete Funktion **showQueue**, die als Hilfsfunktion verwendet werden soll, um den **Queue**-Datentyp als Instanz der **Show**-Typklasse zu deklarieren.

c) Definieren Sie geeignete Gleichheit und Vergleich-Infix-Operatoren für Ihren Queue Datentyp und deklarieren Sie damit den **Queue**-Datentyp als Instanz der **Eq**- und **Ord**-Typklasse.

d) Schreiben Sie geeignete Testfunktionen für alle Funktionen Ihres **Queue**-Datentyps.

Wichtige Hinweise

Damit die **dequeue** Operation auf die Verwendung der (++) Funktion verzichten kann, modellieren Sie Ihre Warteschlange mit Hilfe von zwei Listen. Elemente werden immer aus der ersten Liste entfernt und neue Elemente werden am Anfang der zweiten Liste eingefügt. Wenn die erste Liste leer ist und ein weiteres Element entfernt werden soll, wird die zweite Liste umgedreht und als erste Liste gesetzt.

2. Aufgabe (4 Punkte)

In dem Haskell-Prelude ist die **iterate**-Funktion wie folgt definiert:

```
iterate      :: (a -> a) -> a -> [a]
iterate f x  =  x : iterate f ( f x )
```

Definieren Sie unter Verwendung der **iterate**-Funktion Funktionen, die folgende unendlichen Listen darstellen:

[1, 1, 1, ...]

[1, 2, 3, 4,...]

[2, 4, 6, …]

[0, 1, 3, 7, 15, 31, 63, …]

3. Aufgabe (6 Punkte)

Eine **unfold** Funktion, die ein einfaches rekursives Pattern, um eine Liste zu produzieren, implizit darstellt, kann wie folgt definiert werden.

```
unfold p f g x    | p x         =  []
                  | otherwise = f x : unfold p f g (g x)
```

Redefinieren Sie unter Verwendung der **unfold**-Funktion folgende Funktionen:

(map f), (iterate f) und dec2bin

4. Aufgabe (6 Punkte)

Analysieren sie die Komplexität der Funktionen **makeTree** und **codeTable** aus dem Hoffman-Algorithmus (siehe Vorlesungsfolien).

Wichtige Hinweise:

1) Zu jeder Funktion sollen Testfunktionen geschrieben werden.

2) Verwenden Sie geeignete Namen für Ihre Variablen und Funktionsnamen, die den semantischen Inhalt der Variablen oder die Semantik der Funktionen wiedergeben.

3) Verwenden Sie vorgegebene Funktionsnamen, falls diese angegeben werden.

4) Kommentieren Sie Ihre Programme.

5) Verwenden Sie geeignete lokale Funktionen und Hilfsfunktionen in Ihren Funktionsdefinitionen.

5) Schreiben Sie in allen Funktionen die entsprechende Signatur.