WiSe 2019/2020

Funktionale Programmierung

5.Übungsblatt fürs Tutorium

Prof. Dr. Margarita Esponda

Themen: Funktionen höherer Ordnung, Sortieralgorithmen und Zeitkomplexität.

1. Aufgabe

Betrachten Sie folgende Funktionsdefinition:

```
isSorted :: (Ord a) => (a->a->Bool) -> [a] -> Bool
isSorted  cmp xs = and (zipWith cmp xs (tail XS))
```

Analysieren Sie die Komplexität der Funktion.

2. Aufgabe (8 Punkte)

Analysieren Sie die Komplexität folgender Variante des BubbleSort-Algorithmus. Verwenden Sie für Ihre Analyse die **isSorted** Funktion der 1. Aufgabe.

```
bubbleSort :: (Ord a) => [a] -> [a]
bubbleSort xs  | isSorted (<=) xs = xs
               | otherwise = bubbleSort (moveBubble xs)
                 where
                 moveBubble [] = []
                 moveBubble [x] = [x]
                 moveBubble (x:y:rest) | (<=) x y   = x: moveBubble (y:rest)
                                       | otherwise  = y: moveBubble (x:rest)
```

3. Aufgabe

a) Definieren Sie folgende polymorphe Funktionen für Mengen:

```
makeSet        -- sortiert die Liste und erstellt eine Menge
inSet ::       -- prüft, ob die Zahl in der Menge ist
union ::       -- Vereinigung
subset ::      -- prüft, ob die erste Menge in der zweiten beinhaltet ist.
intersection :: -- Schnittmenge
(\\)           -- berechnet die Differenzmenge
```

b) Analysieren Sie die Komplexität der **union**, **subset** und **intersection** Funktionen.