Fakultät IV: Elektrotechnik und Informatik
Informatik 1 WS 2001/02,
Jähnichen / Flik / Lechtchinsky / Neunast / Tutoren

Aufgabenblatt 3

letzte Aktualisierung: 08. November, 13:05

Ausgabe: *09.11.2001*
Abgabe: *19./20.11.2001* *Prozent:* *100*

Thema: *Rekursionen, Sequenzen*

Mit diesem Aufgabenblatt soll Euer Verständnis für rekursive Programmierung vertieft, und die Benutzung von Sequenzen in Opal eingeführt werden.

1. Aufgabe (Tut): Rekursionsarten

Welche Rekursionstypen gibt es, worin unterscheiden sie sich?

2. Aufgabe (50 Prozent): Sequenzen

Sequenzen sind eine für die funktionale Programmierung unverzichtbare Datenstruktur. Da sie inhärent rekursiv sind, lässt sich an Algorithmen, die mit ihnen arbeiten, rekursive Programmierung sehr gut üben. Sequenzen in Opal können verschiedene Datentypen als Element enthalten (wobei innerhalb einer Sequenz alle Elemente den gleichen Typ haben), deshalb muss man den gewünschten beim Import in "[]" angeben, z.B.:

```
IMPORT Nat ONLY nat
IMPORT Seq[nat] ONLY seq[nat]
```

Neben dem Datentyp `seq[`α`]` kann man aus der Struktur `Seq` noch Funktionen zum Umgang mit Sequenzen importieren. Wichtig sind für die Bearbeitung dieser Aufgabe die Funktionen `<>`, `::`, `<>?`, `::?`, `ft` und `rt` (verwendet für die Lösung dieser Aufgabe keine anderen aus der Struktur `Seq`!).

2.1. Sequenzen, Reduce und Fold, Akkumulatoren (Tut) Implementiert die Funktionen `reduceSum` und `foldDiff`, die alle Elemente einer Sequenz von `nat`s zu einem zusammenfassen. Als Operation für das Reduzieren jeweils zweier Elemente zu einem sollt Ihr "`+`" für das rechtsassoziative `reduceSum`, und "`-`" für das linksassoziative `foldDiff` verwenden. Welches Ergebnis liefert jeweils ein Aufruf `reduceSum(8 :: 1 :: 3 :: <>)` und `foldDiff(8 :: 1 :: 3 :: <>)`?

2.2. Skalarprodukt Schreibt eine Funktion, die das Skalarprodukt zweier als Sequenzen repräsentierter Vektoren berechnet. Das Skalarprodukt zweier Vektor der gleichen Dimension ist definiert als die Summe der Produkte der jeweils korrespondierenden Elemente beider Vektoren. Zum Beispiel ist $(1, 2, 4) \cdot (2, 1, 3) = 1 \cdot 2 + 2 \cdot 1 + 4 \cdot 3 = 16$.

2.3. Horner-Schema Schreibt eine Funktion, die ein Polynom unter Verwendung des Horner-Schemas an einer Stelle x auswertet. Die Koeffizienten des Polynoms sollen der Funktion als Sequenz übergeben werden. Ein Beispiel: Das Polynom $2x^3 - 5x + 3$ sieht als Sequenz so aus: "`2 :: 0 :: -5 :: 3 :: <>`". Um es an der Stelle x auszuwerten, berechnet man das folgende: $(((2) \cdot x + 0) \cdot x - 5) \cdot x + 3$.

2.4. Liste von 1 **bis** n **generieren** Schreibt eine Funktion, die eine Zahl n übergeben bekommt und als Ergebnis eine Sequenz liefert, die die natürlichen Zahlen von 1 bis n enthält (beide inklusive). Wenn 0 übergeben wird, soll die leere Sequenz zurückgegeben werden.

3. Aufgabe (50 Prozent): Sortieren

Zur Lösung dieser Aufgabe sind neben den bereits vorgestellten Funktionen noch weitere aus der Struktur `Seq` sehr hilfreich. Schaut Euch deshalb die Dokumentation zu dieser Struktur gut an! Diese Aufgaben sollen ohne Zuhilfenahme von Opal-Funktionen wie `filter`, `reduce`, `fold`, `map`, `zip`, u.ä. erledigt werden.

3.1. MergeSort (Tut) Implementiert den Sortieralgorithmus Mergesort. Die Funktion `mergeSort` bekommt eine zu sortierende Sequenz natürlicher Zahlen übergeben und liefert eine entsprechend sortierte Sequenz zurück.

3.2. QuickSort Implementiert den Sortieralgorithmus Quicksort in der Funktion `quickSort`, die die gleiche Funktionalität hat wie `mergeSort`.

QuickSort ist ein sehr oft benutztes Sortierverfahren, da es eine einfache Implementation mit (meistens) hoher Sortiergeschwindigkeit verknüpft. Eine übergebene Sequenz wird dabei wie folgt sortiert:

- Wenn die Sequenz leer ist, ist sie schon sortiert.
- Sonst wähle ein Element (genannt Trennelement) der Sequenz aus und teile die anderen Elemente in zwei Sequenzen auf; die eine soll Elemente kleiner dem Trennelement enthalten, und die andere Elemente größer oder gleich dem Trennelement.
- Nun sortiere rekursiv die beiden Teilsequenzen.
- Die sortierte Sequenz ergibt sich aus dem Zusammenfügen der sortierten Teilsequenz mit den kleineren Elementen, dem Trennelement, und der sortierten Teilsequenz mit den größeren/gleichen Elementen.