

Übungen zu Informatik I

Aufgabe 11-1 Größenordnungen (keine Abgabe)

Welche der folgenden Aussagen sind richtig? Geben Sie für jede richtige Aussage einen Beweis und für jede falsche Aussage eine Begründung an, warum die Aussage nicht zutrifft.

- a) $7n^2 + 3n + 5 = O(n)$.
- b) $3n + 4 = O(n^2)$.
- c) $f(n) = O(f(n))$.
- d) $a_p n^p + a_{p-1} n^{p-1} + \dots + a_0 = O(n^p)$
- e) $1^2 + 2^2 + \dots + n^2 = O(n^2)$.

Aufgabe 11-2 Kassensystem (keine Abgabe)

Der SuperMarkt Ludwigshöhe will auf ein neues Kassensystem umsteigen. Sie wurden beauftragt, die dazu nötige Software zu implementieren – natürlich in SML.

Jeder Verkaufsvorgang läuft folgendermaßen ab: Die Barcodes der verkauften Waren werden vom Kassenspersonal mit einem Scanner erfasst, das Kassensystem speichert Barcodes und Stückzahl der verkauften Artikel. Nachdem alle Artikel eingescannt wurden, wird aus diesen Informationen ein Kassenzettel erstellt. Auf dem Kassenzettel wird für jeden verkauften Artikel die Artikelbezeichnung, der Einzelpreis, die verkaufte Stückzahl dieses Artikels und der sich daraus ergebende Gesamtpreis für diesen Artikel ausgedruckt. Ferner wird am Ende des Kassenzettels der Gesamtpreis für den Verkaufsvorgang ausgegeben. Um diese Funktionalität zur Verfügung stellen zu können, ist im Kassensystem eine Tabelle gespeichert, in der jedem Barcode die Artikelbezeichnung und der Preis zugeordnet wird.

- a) Geben Sie eine Signatur *TABELLESig* an, die alle für die im Kassensystem gespeicherte Tabelle notwendigen Operationen beschreibt. Beachten Sie, dass Artikel neu in das Sortiment aufgenommen und aus dem Sortiment entfernt werden können und dass sich die Preise von Artikeln ändern können.
- b) Geben Sie eine Struktur *TABELLE* an, die die Signatur *TABELLESig* implementiert.
- c) Geben Sie eine Signatur *VERKAUFSig* an, die alle für einen Verkaufsvorgang notwendigen Operationen beschreibt.
- d) Geben Sie eine Struktur *VERKAUF* an, die die Signatur *VERKAUFSig* implementiert.

Aufgabe 11-3 Endliche Mengen (5 Punkte)

In der Vorlesung ist die Rechenstruktur *SET* der endlichen Mengen eingeführt worden. In den folgenden Aufgaben können Sie die auf der Homepage der Vorlesung zu findende Datei *set.sml* verwenden. Diese enthält die Definition der Rechenstruktur *SET*.

- a) Definieren Sie eine SML-Funktion *subset* vom Typ $(\text{"a set"}) * (\text{"a set"}) \rightarrow \mathbf{bool}$, die für zwei Mengen *b* und *c* vom Typ *"a set* bestimmt, ob *b* eine Teilmenge von *c* ist.
- b) Definieren Sie eine SML-Funktion *seteq* vom Typ $(\text{"a set"}) * (\text{"a set"}) \rightarrow \mathbf{bool}$, die für zwei Mengen *b* und *c* vom Typ *"a set* bestimmt, ob sie gleich sind.

- c) Definieren Sie eine SML-Funktion *intersect* vom Typ $(\text{"a set} * \text{"a set}) \rightarrow \text{"a set}$ die für zwei Mengen den Durchschnitt der beiden Mengen berechnet.
- d) Für zwei Mengen A und B ist das cartesische Produkt $A \times B$ wie folgt definiert:

$$A \times B := \{(x, y) | x \in A, y \in B\}$$

Die Menge $A \times B$ besteht also aus allen Paaren (x, y) mit $x \in A$ und $y \in B$. Definieren Sie eine SML-Funktion *cartprod* vom Typ $\text{"a set} * \text{"b set} \rightarrow (\text{"a} * \text{"b}) \text{ set}$, die für zwei Mengen A und B vom Typ "a set bzw. "b set das cartesische Produkt $A \times B$ berechnet.

Ihre Funktionen sollen hierbei *ausschließlich* die in der Signatur *SETSig* enthaltenen Operationen zur Konstruktion bzw. Manipulation von Mengen verwenden.

Aufgabe 11-4

Schlangen

(7 Punkte)

Eine wichtige, der in der Vorlesung vorgestellten Datenstruktur der Stapel ähnliche, Datenstruktur heißt *Schlange* (oder *queue*). Im Unterschied zu den Stapeln erfolgt bei Schlangen der Zugriff nach dem Prinzip „First-In-First-Out“ (FIFO). Eine mögliche Vorstellung einer Schlange ist eine Warteschlange - derjenige, der sich zuerst anstellt, wird auch als erstes bedient. Die Grundoperationen auf Schlangen sind die folgenden:

| | |
|-----------------------|---|
| <i>emptyqueue</i> : | die leere Schlange |
| <i>isemptyqueue</i> : | prüft, ob eine Schlange leer ist |
| <i>enter</i> : | hängt („hinten“) ein neues Element an die Schlange an |
| <i>first</i> : | liefert das erste Element einer Schlange |
| <i>dequeue</i> : | entfernt das erste Element einer Schlange |

- a) Geben Sie eine SML-Signatur *QUEUESig* an, die alle oben beschriebenen Operationen enthält.
- b) Geben Sie eine SML-Struktur *QUEUE* an, die sämtliche Operationen der Signatur *QUEUESig* implementiert.
- c) Definieren Sie eine SML-Funktion *posqueue* vom Typ $\text{int queue} \rightarrow \text{int queue}$, so dass ein Aufruf *posqueue*(q) eine Schlange zurückgibt, die alle nichtnegativen Elemente von q in der gleichen Reihenfolge wie in q angeordnet sind.
- d) Definieren Sie eine SML-Funktion *revqueue* vom Typ $\text{'a queue} \rightarrow \text{'a queue}$, die für eine gegebene Schlange q diejenige Schlange berechnet, die die selben Elemente wie q enthält, die aber in umgekehrter Reihenfolge angeordnet sind.
- e) Definieren Sie eine SML-Funktion *foldlqueue* vom Typ

$$(\text{'a} * \text{'b} \rightarrow \text{'b}) \rightarrow \text{'b} \rightarrow \text{'a queue} \rightarrow \text{'b},$$

die die Linksfaltung einer Schlange mit einer Funktion definiert, d. h. ein Aufruf *foldlqueue*(f)(b)((x_1, \dots, x_n)) für eine Funktion f vom Typ $(\text{'a} * \text{'b}) \rightarrow \text{'b}$, ein b vom Typ 'b und (x_1, \dots, x_n) vom Typ 'a queue liefert $f(x_n, f(x_{n-1}, \dots, f(x_1, b) \dots))$. Dabei ist x_1 das erste Element der Schlange (x_1, \dots, x_n) .

Ihre Funktionen sollen hierbei *ausschließlich* die in der Signatur *QUEUESig* enthaltenen Operationen zur Konstruktion bzw. Manipulation von Schlangen verwenden.

Bitte beachten Sie, dass wir Ihre Lösung nur dann korrigieren und bewerten können, wenn sie als *Textdatei* abgegeben wird. Als Lösung ist ein *lauffähiges* SML-Programm abzugeben. Nicht zur Lösung gehörende Bemerkungen sind zwischen die Kommentarzeichen (* und *) einzuschließen. Alle hiervon abweichenden Hausaufgaben werden mit 0 Punkten bewertet.

Abgabe: Dienstag, 14.1.2003, 12:00 Uhr.