Programmierübungen

Wintersemester 2006/2007

9. Übungsblatt
21. Dezember 2006

Abgabe bis Samstag, 13. Januar 23:59 Uhr.

Die Abgabe Ihrer Bearbeitung können Sie im eClaus-System durchführen. Erarbeiten Sie Lösungsideen zu den Aufgaben möglichst in Kleingruppen. Es wird jedoch von Ihnen erwartet, dass jeder Teilnehmer eine eigene Lösung abgibt. Sollten kopierte Quelltexte abgegeben werden, so werden grundsätzlich alle Kopien mit 0 Punkten bewertet. In den Vortragsfolien der Programmierübungen oder im Skript zur Einführung in die Informatik abgedruckte Quelltexte können verwendet werden, müssen aber der Programmierrichtlinie entsprechend formatiert und kommentiert werden.

Beachten Sie die Programmierrichtlinie und kommentieren Sie Ihren Quelltext. Dokumentieren Sie unbedingt Ihre Lösungsidee in den Quelltext-Kommentaren.

`http://www.iste.uni-stuttgart.de/ps/Lehre/WS0607/inf-prokurs`

Am Montag 8.1.2007, 8:00 Uhr in V38.01 wird das Aufgabenblatt kurz vorgestellt und Sie haben Gelegenheit Fragen zu den Aufgaben zu stellen.

Aufgabe 9.1: Stacks (4 Punkte)

Auf der Webseite zu den Programmierübungen ist das Paket Expression_Trees verfügbar. Laden Sie dieses herunter und erzeugen Sie ein weiteres Paket Node_Stacks. Implementieren Sie in diesem Paket einen abstrakten Datentyp Stack, der Werte des Typs Expression_Trees.Expression_Tree in LIFO-Reihenfolge verwaltet. Implementieren Sie die üblichen Operationen wie im Skript zur Einführung in die Informatik angegeben.

Hinweis: Die Implementierung eines Stacks lässt sich direkt aus einer einfach verketteten Liste ableiten. Verwenden Sie Quelltext aus dem vorigen Übungsblatt wieder.

Aufgabe 9.2: Bäume mit varianten Records (7 Punkte)

Arithmetische Ausdrücke können in der Umgekehrten Polnischen Notation (UPN) geschrieben werden. Dabei werden zunächst die Operanden und danach die Operation notiert, z. B. `2 3.4 + 5 *` für $(2 + 3.4) \cdot 5$. Die Berechnung von Ausdrücken in UPN lässt sich sehr leicht realisieren, indem folgender Algorithmus verwendet wird:

- Verwende einen Stack als Datenstruktur; lese den Ausdruck von links nach rechts.
- Wird ein Operand angetroffen, dann lege diesen Operand auf den Stack.
- Wird eine Operation (`+`, `-`, `*`, `/`) angetroffen, dann nimm zwei Operanden vom Stack, führe die Operation aus und lege das Ergebnis wieder auf den Stack.
- Am Ende der Berechnung ist der Ausdruck vollständig gelesen und es liegt genau eine Zahl auf dem Stack. Diese ist das Ergebnis.

Verwenden Sie Ihr Paket Stacks aus Aufgabe 9.1 um die Prozedur Parse_Postorder zu implementieren. Diese Prozedur soll eine Variante des obigen Algorithmus implementieren, um einen „Rechenbaum“ für den Ausdruck in UPN aufzubauen. Achten Sie darauf, dass Ihre Implementierung auch im Fehlerfall nicht mehr benötigten Speicher korrekt freigibt.

In einem Rechenbaum sind zwei verschiedene Arten von Knoten enthalten: Innere Knoten, die Rechenoperationen entsprechen, und Blätter, die Zahlenwerte modellieren. In dieser Aufgabe wird diese Eigenschaft durch den varianten Record Node modelliert. Dieser Record-Typ besitzt eine Diskriminante Kind, die die Art eines Knotens beschreibt. In Abhängigkeit von dem Werts der Diskriminante besitzt ein Objekt des Typs Node entweder die Komponenten Left und Right oder die Komponente Value.

Hinweise:

- Variante Records haben Sie in der Vorlesung „Einführung in die Informatik 1“ kennen gelernt (Kapitel 1.12.2 im Skript).
- Implementieren Sie die Prozedur Parse_Postorder in mehreren Iterationen. Lassen Sie eine erste Version die einzelnen Rechnenschritte ausgeben und überzeugen Sie sich von der Korrektheit, bevor Sie den Baum aufbauen lassen.
- Im Grundstudiumspool gibt es unter Linux das Programm `dc` mit dem Ausdrücke in UPN ausgerechnet werden können. Verwenden Sie das Kommando
  `dc -e"5k <expr> fq"`
  wobei `<expr>` durch den Ausdruck in UPN ersetzt werden muss. Es wird am Ende der gesamte Stack-Inhalt ausgegeben. Ist dies mehr als eine Zahl, so war der Ausdruck fehlerhaft.

Aufgabe 9.3: Baumtraversierung (9 Punkte)

Implementieren Sie die weiteren Unterprogramme des Pakets Expression_Trees: In der Prozedur Put_Preorder soll der Ausdruck in Funktionsnotation ausgegeben werden, in Put_Inorder soll die Operator-Notation erzeugt werden. Es sollen möglichst wenige Teilausdrücke geklammert werden. Die Funktion Value soll die durch einen Rechenbaum spezifizierte Berechnung ausführen und das Resultat zurückgeben. Die Prozedur Destroy soll den Speicher für einen Baum freigeben.

Schreiben Sie ein Hauptprogramm Translator, das den Benutzer zur Eingabe eines Ausdrucks in UPN auffordert und daraufhin diesen Ausdruck nach Preorder- sowie nach Inorder-Notation übersetzt und jeweils das Ergebnis der Berechnung angibt.

Lassen Sie Zahlen mit einer Genauigkeit von Ada.Float_Text_IO.Default_Aft Stellen nach dem Komma ausgeben.

Beispiele:

```
Ausdruck in UPN: 5.4 3.6 + 8 - 10 * 20 * 3 -
Preorder: "-" ("*" ("*" ("-" ("+" (5.40000, 3.60000), 8.00000),
 10.00000), 20.00000), 3.00000) = 197.00000
Inorder: (5.40000 + 3.60000 - 8.00000) * 10.00000 * 20.00000 -
3.00000 = 197.00000
```