Institut für Informatik
Dr. Martin Buchholz
Benjamin Peherstorfer

Technische Universität München
WS 2010/2011

Einführung in die wissenschaftliche Programmierung

Übungsblatt 9

1.) Taschenrechner

In dieser Aufgabe soll ein Taschenrechner erstellt werden. Als Operationen sind nur + und ∗ zugelassen, dafür soll er mit Klammerausdrücken umgehen können.

Grundsätzliches Vorgehen: Das Programm soll zuerst die Eingabe (in Infix-Notation), z.B.

```
( 5 * ( ( ( 9 + 8 ) * ( 4 * 6 ) ) + 7 ) )
```

in die Postfix-Notation umwandeln, d.h.

```
5 9 8 + 4 6 * * 7 + *
```

In der Postfix-Notation steht der Operator nach den Operanden, d.h. `( 4 + 3 )` wird zu `4 3 +`. Der Vorteil ist, dass keine Klammern mehr benötigt werden und sich dann der Ausdruck in Postfix-Notation einfacher auswerten lässt.

Eine nützliche Datenstruktur für die Auswertung dieser Ausdrücke ist ein *Stack* oder *Stapel*. In Python kann ein Stack mit einer Liste dargestellt werden. Dabei entspricht die Operation `push` der Operation `append`. Die Operation `pop` ist bereits eingebaut.

Implementierung: Schreiben Sie eine Funktion `infix2postfix` welche einen Ausdruck (als String) in Infix-Notation in die Postfix-Notation umwandelt. Verwenden Sie einen Stack, und durchlaufen Sie den Ausdruck. Ist das aktuelle Zeichen

i) ein Operator, legen Sie “+” oder “*” auf den Stack,

ii) eine rechte Klammer “)”, dann nehmen Sie ein Element vom Stapel und hängen es an den Postfix-Ausdruck an,

iii) eine Zahl, so hängen Sie diese (als String!) an den Postfix-Ausdruck an,

Leerzeichen und linke Klammern werden ignoriert. Sie können davon ausgehen, dass Sie einen gültigen Ausdruck bekommen, dass Leerzeichen zwischen Operator, Operanden und Klammern sind, und dass nur natürliche Zahlen kleiner 10 vorkommen.

Die Auswertung des Postfix-Ausdrucks funktioniert ganz ähnlich. Verwenden Sie wieder einen Stack und durchlaufen Sie den String Element für Element. Ist das aktuelle Zeichen

i) ein Operator, holen Sie die beiden letzten Elemente vom Stapel (zweimal `pop`), wenden Sie den Operator an und legen Sie das Ergebnis wieder auf den Stapel,

ii) eine Zahl, legen Sie diese auf den Stapel.

Nach diesem Durchlauf, liegt das Ergebnis als einziges Element am Stack.

2.) AVL Bäume

In der Vorlesung wurden AVL Bäume eingeführt. Implementieren Sie eine `AVLBaum` Klasse, welche das Traversieren und das Einfügen von Knoten erlaubt. Sie benötigen dazu die Einfach- und die Doppelrotation.

Als Schlüssel sollen nur natürliche Zahlen erlaubt sein. Bei der Traversierung des Baums sollen die Schlüssel in aufsteigender Reihenfolge ausgegeben werden.