

Übungsblatt 10 **Ausgabe: Mi, 18.07.00** **Abgabe: keine, 18 Uhr**

Hinweis: Dieses Übungsblatt wird von den Tutoren nicht korrigiert und geht auch nicht in die Punktwertung ein. Der behandelte Stoff ist aber trotzdem relevant für die Klausur.

Aufgabe 1: Parsing [10 Punkte]

Gegeben ist folgende Grammatik für einfache arithmetische Ausdrücke in Infixnotation:

$$\begin{aligned} \text{expr} &::= \text{term} \mid \text{term} + \text{expr} \mid \text{term} - \text{expr} \\ \text{term} &::= \text{factor} \mid \text{factor} * \text{term} \mid \text{factor} / \text{term} \\ \text{factor} &::= (\text{expr}) \mid \text{NUM} \mid \text{ID} \\ \text{NUM} &::= '0' \mid \dots \mid '9' \\ \text{ID} &::= 'a' \mid \dots \mid 'z' \end{aligned}$$

Dabei ist *expr* das Startsymbol. Als Beispiel dient der Ausdruck $5 * a + b + c / (d + 3)$

- (a) [4 Punkte] Zeichnen Sie den Parsetree.
- (b) [1 Punkt] Ist die Grammatik eindeutig?
- (c) [2 Punkte] Ergänzen Sie den Parsetree um semantische Aktionen zur Erzeugung eines Postfix-Ausdrucks. Wie sieht der von ihrem Parsetree erzeugte Postfix-Ausdruck aus?
- (d) [3 Punkte] Zu dieser Grammatik soll nun ein Parser in C entwickelt werden. Da die Symbole nur ein Zeichen lang sind, ist keine lexikalische Analyse notwendig. Als Ausgabe soll ein äquivalenter Ausdruck in Postfix-Form generiert werden.
Hinweise:
 - Jedem Nichtterminal entspricht genau eine Prozedur
 - Der Parser kann ohne Backtracking arbeiten
 - Die Ausgabe sollte nur über eine einzige Prozedur stattfinden (d.h. die write-Anweisungen sollten nicht verstreut im ganzen Programm liegen).
 - Da der Parser gleichzeitig einen Postfix-String erzeugen soll, müssen einige der Prozeduren semantische Aktionen in Form von Ausgabeanweisungen enthalten.
 - Bei einem Fehler in der Eingabe soll das Programm sofort verlassen werden.

Aufgabe 2: Codeerzeugung [10 Punkte]

Der Parser aus Aufgabe 2d) soll nun so modifiziert werden, daß er Code für den 68000-Assembler erzeugt. Die arithmetischen Ausdrücke sollen allerdings keine Variablen mehr enthalten (d.h. es sind nur noch Ziffernkonstanten zulässig). Ändern Sie die Ausgaberroutine aus Aufgabe 2d) entsprechend ab.

Hinweise:

- Die Assembleranweisungen sollen den Stack des 68000'er verwenden.
- Das Ergebnis des Ausdrucks soll im Register *D0* stehen.