34 Punkte

Aufgabe 1 (Lex und Yacc)

Ziel dieser Aufgabe ist es, mit Hilfe von Lex und Yacc römische Zahlen in Dezimalzahlen umzuwandeln. Bekanntermaßen stehen im römischen Zahlensystem die Ziffern (bzw. Buchstaben) I für 1, V für 5, X für 10, L für 50, C für 100, D für 500 und M für 1000 zur Verfügung. Die Zeichen können unter Beachtung bestimmter Regeln nebeneinander geschrieben werden. Die Regeln lauten wie folgt:

R1: Die Zeichen I, X, C und M dürfen wiederholt nebeneinander stehen, die Zeichen V, L und D jedoch nicht.

R2: Stehen gleiche Zeichen nebeneinander, so werden sie addiert (z.B. XXX = 30 = 10+10+10). Jedoch dürfen die Zeichen I, X und C nicht mehr als dreimal nebeneinander auftreten. Das Zeichen M kann beliebig oft nebeneinander angegeben werden.

R3: Die Zeichen V, L und D dürfen in einer Zahl nur einmal vorkommen.

R4: Steht ein Zeichen für eine kleinere Einheit rechts neben einem Zeichen für eine größere Einheit, dann wird die kleinere Einheit auf die größere Einheit addiert (z.B. DCCLVI = 756 = 500+100+100+50+5+1).

R5: Steht ein Zeichen für eine kleinere Einheit links neben einem Zeichen für eine größere Einheit, dann wird die kleinere Einheit von der größeren Eineheit subtrahiert (z.B. XXIX = 29 = 10+10+10-1).

R6: Es dürfen nicht zwei oder mehr kleinere Einheiten von einer rechts stehenden größeren Einheit abgezogen werden.

Die Darstellung einer Dezimalzahl durch eine römische Zahl ist nicht eindeutig. Es gilt z.B. MIM = MCMIC = MCMXCIX = 1999.

12 Punkte

(a) Schreiben Sie ein Lex-Programm, das eine Folge von durch Zeilenendezeichen getrennten römischen Zahlen als Eingabe entgegen nimmt, die Eingabe lexikalisch analysiert und Vorkehrungen für die Syntaxanalyse mit Yacc trifft. Beachten Sie, daß auch eine Kleinschreibung der römischen Zahlen erlaubt ist.

22 Punkte

(b) Schreiben Sie ein Yacc-Programm, das die Eingabe syntaktisch analysiert, bei korrekter Syntax die entsprechende Dezimalzahl berechnet, bei Verletzung der Regel R3 anstelle der Berechnung eine Warnung ausgibt und ansonsten einen Syntaxfehler meldet.

Aufgabe 2 (Operator-Vorrang-Analyse) *34 Punkte*

Gegeben sei die Grammatik $G = (N, \Sigma, P, E)$ mit

$$P = \{\ E \to EAE \mid (E) \mid \mathbf{id},$$
$$A \to + \mid * \mid \uparrow \}$$

(a) Wörter aus L(G) können als Terme einer Integer-Algebra interpretiert werden. Formulieren Sie eine zu G äquivalente Grammatik G', die nur *ein* Nichtterminal enthält. Konstruieren Sie dann die Tabelle für eine Operator-Vorrang-Analyse zu G'. Die Prioritäten und Assoziativitäten der in G' enthaltenen Operatoren sollen dabei den üblichen Regeln für die gleichnamigen Integer-Operatoren entsprechen. '↑' ist der Potenzoperator. *12 Punkte*

(b) Parsen Sie mit Hilfe der in (a) definierten Tabelle die Folge *12 Punkte*

$$\mathbf{id} + \mathbf{id} \uparrow (\mathbf{id} * \mathbf{id})$$

(c) Das platzraubende Abspeichern der Tabelle kann durch geschickte Wahl zweier Vorrangfunktionen *10 Punkte*

$$f, g : \sum \cup \{\$\} \to \mathbb{N}_0$$

vermieden werden. Sofern in der Vorrangtabelle die Position (op1, op2) besetzt ist, soll gelten:

$op1 <\cdot op2 \Leftrightarrow f(op1) < g(op2)$

$op1 \doteq op2 \Leftrightarrow f(op1) = g(op2)$

$op1 \cdot> op2 \Leftrightarrow f(op1) > g(op2)$

Um die Vorrangrelation zwischen zwei Operatoren zu ermitteln, wird nun nicht mehr in der Tabelle nachgesehen, sondern die Ergebnisse der Aufrufe von f und g werden miteinander verglichen.

Definieren Sie zwei geeignete Funktionen f und g für die Tabelle aus (a).

17 Punkte **Aufgabe 3 (LL(1)-Grammatiken)**

Gegeben sei die folgende Grammatik $G = (N, \Sigma, P, S)$ mit

$$
\begin{aligned}
N &= \{ S, E, L \} \\
\Sigma &= \{ \mathbf{id}, \mathbf{print}, \mathbf{num}, ;, ,, (,), :=, + \} \\
P &= \{ S \rightarrow S;S \mid \mathbf{id} := E \mid \mathbf{print}(L), \\
&\quad E \rightarrow \mathbf{id} \mid \mathbf{num} \mid E + E, \\
&\quad L \rightarrow E \mid L, E \}.
\end{aligned}
$$

Überführen Sie G in eine äquivalente LL(1)-Grammatik G'.

40 Punkte **Aufgabe 4 (FIRST- und FOLLOW-Mengen)**

Gegeben sei die folgende LL(1)-Grammatik $G = (N, \Sigma, P, S)$ mit

$$
\begin{aligned}
N &= \{ S, A, B, C, D, E, F, G \}, \\
\Sigma &= \{ \mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, (,), ;, : \} \text{ und} \\
P &= \{ \\
&\quad (1)\ S \rightarrow \mathbf{c}\,\mathbf{a}\,A; \\
&\quad (2)\ A \rightarrow (B) \\
&\quad (3)\ A \rightarrow \varepsilon \\
&\quad (4)\ B \rightarrow DE \\
&\quad (5)\ B \rightarrow C \\
&\quad (6)\ C \rightarrow \mathbf{a}:\mathbf{b}F \\
&\quad (7)\ F \rightarrow ;\mathbf{a}:\mathbf{b}F \\
&\quad (8)\ F \rightarrow \varepsilon \\
&\quad (9)\ D \rightarrow \mathbf{d}\,\mathbf{a}:\mathbf{b};G \\
&\quad (10)\ G \rightarrow D \\
&\quad (11)\ G \rightarrow \varepsilon \\
&\quad (12)\ E \rightarrow C \\
&\quad (13)\ E \rightarrow \varepsilon \\
&\}
\end{aligned}
$$

17 Punkte (a) Berechnen Sie die initialen Steuermengen. Erstellen Sie zunächst gemäß dem im Kurs angegebenen Verfahren einen Graphen, der die Reihenfolge zur Berechnung der FIRST-Mengen angibt.

17 Punkte (b) Bestimmen Sie die Steuermengen. Berechnen Sie dazu die FOLLOW-Mengen mit dem im Kurstext beschriebenen Algorithmus und geben Sie auch den Graphen an, der aus der Anwendung dieses Algorithmus resultiert.

(c) Stellen Sie die Analysetafel auf. *6 Punkte*

Aufgabe 5 (Schleifenoptimierung) *25 Punkte*

Gegeben sei das folgende Programmstück:

```
FOR i := 1 TO k1 DO
  FOR j := 1 TO k2 DO
    a := n * i;
    b := j * log(n);
    DoSomething(a,b);
  END
END
```

(a) Führen Sie auf der Quellcode-Ebene die möglichen *Verlagerungen von Schleifeninvarianten* durch. *7 Punkte*

(b) Übersetzen Sie das Ergebnis aus (a) in 3-Adreß-Code (3AC). Gehen Sie dabei davon aus, daß es sich bei `DoSomething(x, y)` um eine legale 3AC-Anweisung handelt. *5 Punkte*

(c) Bestimmen Sie die Basisblöcke und den Flußgraphen zum 3AC-Programm aus (b). *8 Punkte*

(d) An zwei Stellen im 3AC-Programm aus (b) lassen sich *Berechnungen mit Schleifenvariablen* vereinfachen, indem jeweils eine Multiplikation in eine Addition überführt wird. Führen Sie *eine* der beiden möglichen Vereinfachungen durch. *5 Punkte*