Hinweise zur Bearbeitung der Klausur zum Kurs 1810 „Übersetzerbau“

Bitte lesen Sie sich diese Hinweise vollständig und aufmerksam durch, bevor Sie mit der Bearbeitung der Klausur beginnen.

1. Die Klausurdauer beträgt 3 Stunden.
2. Prüfen Sie bitte die Vollständigkeit Ihrer Unterlagen. Die Klausur umfaßt
   - 2 Deckblätter
   - diese Hinweise
   - 1 Formblatt für eine Teilnahmebescheinigung zur Vorlage beim Finanzamt
   - 4 Aufgaben auf den Seiten 2 - 4
3. Bevor Sie mit der Bearbeitung der Klausuraufgaben beginnen, füllen Sie bitte die folgenden Teile der Klausur aus:

   (a) *sämtliche* Deckblätter mit Name, Anschrift sowie Matrikelnummer. Markieren Sie vor der Abgabe auf allen Deckblättern die von Ihnen bearbeiteten Aufgaben.

   (b) die Teilnahmebescheinigung, falls Sie diese wünschen.
4. Schreiben Sie Ihre Lösungen auf Ihr eigenes Papier (DIN A4) und nicht auf die Seiten mit den Aufgabenstellungen. Heften Sie vor Abgabe der Klausur die Deckblätter, die Klausuraufgaben und ggf. die Teilnahmebescheinigung an Ihre Bearbeitung.
5. Schreiben Sie bitte auf jedes Blatt oben links Ihren Namen und oben rechts Ihre Matrikelnummer. Numerieren Sie Ihre Seiten bitte durch.
6. Als Hilfsmittel sind nur unbeschriebenes Konzeptpapier und Schreibzeug (kein Bleistift!) zugelassen.

Es sind maximal 100 Punkte erreichbar. Wenn Sie mindestens 50 Punkte erreicht haben, können Sie davon ausgehen, daß Sie die Klausur bestanden haben.

Aufgabe 1 (Endliche Automaten) *20 Punkte*

Geben Sie für jede der folgenden Sprachen einen deterministischen, endlichen Automaten an, der diese akzeptiert. Verwenden Sie für die Darstellung der Übergangsfunktionen Zustandsübergangsdiagramme.

(a) $L_1 = \{ w \in (0,1)^* \mid w$ die Anzahl der Einsen in w ist ungerade und die Anzahl der Nullen in w ist gerade$\}$ *6 Punkte*

(b) $L_2 = \{ w \mid w$ stellt eine Dezimalzahl dar, für die gilt: $v(w)$ % $5 = 3\}$, wobei $v(w)$ den Wert der Dezimalzahl und % die Modulo-Funktion beschreibt *4 Punkte*

(c) $L_3 = \{ w \mid w$ stellt eine durch 3 teilbare Binärzahl dar$\}$ *10 Punkte*

In den Zahlendarstellungen sind führende Nullen erlaubt, wogegen das leere Wort keine gültige Zahl darstellt. Die Null ist als gerade Zahl zu interpretieren.

Aufgabe 2 (LL(1)-Grammatik) *25 Punkte*

Gegeben sei die folgende, kontextfreie Grammatik
$G = (\{S, A, B, C\}, \{\mathbf{a}, \mathbf{c}, \mathbf{f}, \mathbf{g}, \mathbf{h}\}, P, S)$,

mit $P = \{$ $S \rightarrow ABA$
$A \rightarrow \mathbf{a}Ad \mid A\mathbf{ac} \mid A\mathbf{aa} \mid \mathbf{g}$
$B \rightarrow \mathbf{fc} \mid C \mid B\mathbf{h}$
$C \rightarrow \mathbf{c} \mid CC$ $\}$.

(a) Geben Sie eine äquivalente Grammatik G' mit LL(1)-fähigen Produktionen an, indem Sie Linksrekursion entfernen und ggf. eine Linksfaktorisierung durchführen. *8 Punkte*

(b) Berechnen Sie die FIRST- und FOLLOW-Mengen Ihrer Grammatik G'. *10 Punkte*

(c) Geben Sie die Steuermengen der Produktionen von G' an.
Ist G' eine LL(1) Grammatik? Begründen Sie Ihre Antwort. *7 Punkte*

Aufgabe 3 (LR-Analyse) *35 Punkte*

Gegeben sei die Grammatik $G = (\{S, E\}, \{\textbf{if}, \textbf{then}, \textbf{else}, \textbf{a}, \textbf{b}, \textbf{c}, \textbf{d}\}, P, S)$,

mit $P = \{$ (1) $S \rightarrow$ **if** E **then** S
(2) $S \rightarrow$ **if** E **then** S **else** S
(3) $S \rightarrow$ **c**
(4) $S \rightarrow$ **d**
(5) $E \rightarrow$ **a**
(6) $E \rightarrow$ **b** $\}$.

(a) Wieso ist diese Grammatik mehrdeutig? *3 Punkte*

(b) Berechnen Sie die kanonische LR(0)-Kollektion für G. *15 Punkte*

(c) Erstellen Sie eine Analysetabelle. Beheben Sie den auftretenden *shift/reduce*-Konflikt so, daß ein kompletter „**if ... then ... else**“ Ausdruck beim Parsen erkannt wird. *10 Punkte*

(d) Analysieren Sie den Ausdruck „**if a then if b then c else d**“. *7 Punkte*

Aufgabe 4 (Code-Optimierung) *20 Punkte*

Führen Sie auf dem angegebenen Programmstück der Reihe nach die folgenden Optimierungsschritte durch. Geben Sie nach jedem Schritt an, wie das Programm verändert wurde (d.h. schreiben Sie die geänderten Zeilen auf). Geben Sie das Programm nach dem letzten Optimierungsschritt komplett an.

Optimierungsschritte:

1. Elimination toten Codes
2. Konstantenpropagation und Konstantenfaltung
3. Schleifenfaltung (und ggf. nochmal Schritt 2)
4. Verlagerung von Schleifeninvarianten
5. Code-Hoisting

Programmstück:

```
(1)  m := 100;
(2)  f := false;
(3)  p := 1;
(4)  y := p+p;
(5)  z := 0;
(6)  for i := 0 to y do begin
(7)    z := z+p
(8)  end;
(9)  if f then begin
(10)   for x := y to z do begin
(11)     i := i+x;
(12)     if i > y then f := true
(13)   end
(14) end;
(15) if not f then begin
(16)   if x < k then
(17)     x := k+x+1
(18)   else
(19)     if x > k then
(20)       x := k+x-1
(21)     else
(22)       if x = k then
(23)         x := sqrt(k+x)
(24)       else
(25)         if f then
(26)           x := k
(27)         end
(28)       end
(29)     end
(30)   end
(31) end;
(32) for i := 0 to m do begin
(33)   for j := i to m do begin
(34)     z := z+n*i+sqrt(m)-x
(35)   end
(36) end;
```