# Übung 11 zum Compilerbau WS14/15 bis: 20.01.2015

**Aufgabe 55:**
Gegeben sei folgender Block:
I = {A,B,C}, O = {F,G} mit folgendem Anweisungsteil:

```
S1:   T ← A + B;
S2:   S ← A - B;
S3:   F ← T * S;
S4:   T ← A - B;
S5:   S ← A - C;
S6:   R ← B - C;
S7:   T ← T * S;
S8:   G ← T * R;
```

**a)** Welche Werte liefert obiger Block?

**b)** Verbessern Sie obigen Block durch Anwendung von Transformationen, dabei ist Flipping nicht zu verwenden.

**c)** Bringen Sie den Block aus b) durch Flipping in eine Form, die einem möglichst effizienten Assemblerprogramm entspricht.

**Aufgabe 56:**
Geben Sie jeweils die Transformationsschritte an, mit denen man vom Programm auf Folie „Codeverbesserung" 19 zum Programm auf Folie „Codeverbesserung" 20 gelangt.

**Aufgabe 57:**
Bei einer Akkumaschine haben die Befehle jeweils nur eine Adresse. Der zweite Operand ist immer der Akku und das Ergebnis wird auch im Akku abgelegt. Die Befehle lauten wie die der Z-Code-Maschine. Dabei wird jedoch nicht auf dem Datenkeller operiert, sondern auf dem Akku. Will man z.B. die Anweisung Y← X * Z programmieren, so macht man das auf der Akkumaschine wie folgt:

```
LOAD X;     //Akku = X
MULT Z;     //Akku = Akku * Z
STORE Y;    //Y = Akku;
```

**a)** Geben Sie ein Programm der Akkumaschine zur Auswertung des Baumes auf Folie „Codeverbesserung 19" an.

**b)** Geben Sie ein Programm der Akkumaschine zur Auswertung des Baumes auf Folie „Codeverbesserung 20" an.

**Aufgabe 58:**
Geben Sie jeweils die Akkumaschinenprogramme für die Blöcke aus Aufgabe 55 b) und c) an und vergleichen Sie die Anzahl der Befehle

# Übung 11 zum Compilerbau WS14/15 bis: 20.01.2015

**Aufgabe 44:**
Seien $\Sigma_3 = \{1,2,3\}$ als Alphabet gegeben und L die Sprache über $\Sigma_3$, die alle Wörter enthält, in denen keine Ziffer zweimal direkt hintereinander auftritt. L enthält z.B. folgende Wörter:
1232123, 32313231, ε,... , aber nicht 23112 oder 211,...

**a)** Geben Sie einen NEA mit maximal 5 Zuständen an, der L erkennt. Dabei dürfen Sie Transitionen mit alternativen Möglichkeiten beschriften.

**b)** Führen Sie das NEA-Verfahren mit dem NEA aus a) und den Wörtern 12321 und 112 durch.

**Aufgabe 45:**
Gegeben seien das Alphabet $\Sigma_{bool} = \{0, 1, (, ), and, or, not\}$. Die Sprache $L_{bool}$ sei die Sprache der booleschen Ausdrücke, wobei die Operatoren `and` und `or` in Infixnotation und `not` in Präfixnotation geschrieben wird.
Somit sind folgende Zeichenketten korrekte boolesche Ausdrücke:
`(0 and 1) or (not 1)` oder (ohne Klammerung) `0 and 1 or not 1`

**a)** Geben Sie eine kontextfreie Grammatik mit maximal 6 Regeln an, die $L_{bool}$ erzeugt.

**b)** Geben Sie mindestens zwei Syntaxbäume für den booleschen Ausdruck `0 and 1 or not 1` an, die verschiedene Linksableitungen repräsentieren.

**c)** Welcher der Syntaxbäume aus b) würde dem geklammerten Ausdruck entsprechen?

**d)** Geben Sie zu jedem der Syntaxbäume aus b) die zugehörige Linksableitung an und führen Sie dabei die jeweils verwendete Regel auf.

**Aufgabe 46:**
Gegeben sei folgende kontextfreie Grammatik:

G = ({S,L}, {'a', '(' , ')' , ','}, S, P)

mit P = {
S -> ( L ) (1)
S -> a (2)
L ->L , S (3)
L -> S (4) }

**a)** Geben Sie jeweils ein Wort an, das durch G erzeugbar ist und ein, zwei oder drei a's enthält.

**b)** Geben Sie eine erfolgreiche Rechnung des nichtdeterministischen Top-Down-Analyseautomaten zu obiger Grammatik und dem Wort (a,a) an.

**c)** Geben Sie den bei der Ableitung aus b) konstruierten Syntaxbaum an.

**d)** Geben Sie eine nicht erfolgreiche Rechnung des nichtdeterministischen Top-Down-Analyseautomaten zu obiger Grammatik und dem Wort (a,a) an.

# Übung 11 zum Compilerbau WS14/15 bis: 20.01.2015

**Aufgabe 47:**
Gegeben sei folgendes Mini-Java-Programm **P**:

```
final int in = 4;
int i, out = 1;
{
      i = in;
      while i > 1 {
            out = out * i;
            i = i - 1;
      }
      print(out);
}
```

Berechnen Sie die Semantik von **P**. Geben Sie hierzu folgendes an:

**a)** die Umgebung und den Zustand nach dem Deklarationsteil,

**b)** den Zustand jeweils vor jeder Berechnung der Schleifenbedingung,

**c)** die Ausgabe (auch wenn diese eigentlich nicht zur Semantik gehört).

**Aufgabe 48:**
Gegeben sei folgendes Mini-Java-Programm **P**:

```
final int in = 3;
int i, out = 1;
{
            i = in;
            while i > 1 {
                  out = out * i;
                  i = i - 1;
            }
            print(out);
}
```

**a)** Geben Sie die Übersetzung von **P** in Zwischencode an. Führen Sie hierzu die Symboltabelle, den Wert des Hauptspeichers und den Z-Mini-Java-Code auf.

**b)** Führen Sie eine Rechnung des Z-Mini-Java-Codes aus a) beginnend mit dem in a) berechneten Wert des Hauptspeichers durch.
Geben Sie dabei den Befehlszähler und den Datenkeller bei jedem Schritt an und den Hauptspeicher nur, wenn er sich ändert.