**Saarland University**
**Department 6.2 - Computer Science**
**Prof. Dr. W. J. Paul**
**Jonas Oberhauser**

# System Architecture - SS15

**Exercise Sheet 8(due: June 15, 2015)**

---

**Wichtig:**

- Sie Benötigen 50% aller Übungsblätter die für Klausur X relevant sind, um zu Klausur X zugelassen zu werden. Dieses Blatt ist Relevant für Haupt- und Nachklausur.
- Das Übungsblatt muss stets am Montag nach der Vorlesung bei mir in der Office Hour oder, falls zeitgleich, in der Übungsgruppe Ihrer Tutorin abgegeben werden.
- Geben Sie stets Ihren Namen, Ihre Matr. Nr., und den Namen ihrer Tutorin auf der vordersten Seite oben rechts an.
- Sie dürfen Ergebnisse von vorherigen Aufgaben verwenden, auch wenn Sie diese nicht gelöst haben. Markieren sie Gleichungen, in denen Sie ein vorheriges Ergebniss benutzen, mit dem Kürzel E+Aufgabenblatt+Aufgabennummer.
- Wenn Sie sich nicht für die Klausur vorbereiten möchten, aber trotzdem zugelassen werden möchten, schreiben Sie einfach Ihren Namen und Ihre Matrikelnummer auf die Lösung einer kompetenten Mitstudentin. Es besteht auch keine Anwesenheitspflicht in den Übungsgruppen.

Tutor: _____

Namen, Matr. Nummern: _____

## Bonus Aufgabe 1: (3)

Nennen Sie 3 *wichtige* Unterschiede zwischen C und C0!

> **Solution:** Keine Addressarithmetik, keine dynamischen Arrays, Garbage Collection (bzw. kein free)

## Aufgabe 2: (3)

Definieren Sie in C0 Syntax (ableitbar mit $\langle TyDS \rangle$) folgende Typen. Beachten Sie die Kontextbedingungen.

(a) [1 point] Binäre Bäume `TREE`.

(b) [1 point] Doppelt Verlinkte Listen `LEL` vom Typ `int`. D.h. für $s \in ra(\texttt{LEL})$ soll

$$s(\texttt{val}) \in ra(\texttt{int}).$$

(c) [1 point] Eine $5 \times 8$ Matrix `Matrix_5_8` vom Typ `char`. D.h. für $(i, j) \in [0:4] \times [0:7]$ und $s \in ra(\texttt{Matrix\_5\_8})$ soll

$$s(i)(j) \in ra(\texttt{char}).$$

> **Solution:** typedef TREE* TREEp ; typedef TREEp[2] TREE
>
> typedef LEL* LELp ; typedef struct  LELp last; int val; LELp next  LEL
>
> typedef char[8] Col ; typedef Col[5] Matrix

**Saarland University**
**Department 6.2 - Computer Science**
**Prof. Dr. W. J. Paul**
**Jonas Oberhauser**

# System Architecture - SS15

**Exercise Sheet 8(due: June 15, 2015)**

---

## Aufgabe 3: (6)

Betrachten Sie folgendes unvollständiges Programm:

```
typedef DTE* DTEp;
typedef struct {
    uint label;
    DTEp father;
    DTEp fson;
    DTEp bro
} DTE;

typedef SEQ* SEQp;
typedef struct {DTEp _dte; SEQp next} SEQ;

DTEp programtree;

DTEp nson(DTEp p, uint j)
{
    DTEp result;
    p = p*.fson;
    while p*.bro!=null && j!=0
    {
        p = p*.bro;
        j = j - 1
    };
    if j!=0 {result = null} else {result = p};
    return result
}


SEQp fseq(DTEp seq)
{
    SEQp current;

}

DTEp se(DTEp seq, uint index)
{


}

uint main ()
{
    ...
    return 0
}
```

**Saarland University**
**Department 6.2 - Computer Science**
**Prof. Dr. W. J. Paul**
**Jonas Oberhauser**

# System Architecture - SS15

**Exercise Sheet 8(due: June 15, 2015)**

(a) [4 points] Bestimmen Sie die Typtabelle $tt$ und die Funktionstabelle $ft$.

(b) [2 points] Geben Sie für alle Top-Level Typen zwei verschiedene Instanzen des Typs an, i.e., $u, v \in ra(T)$ für alle $T \in ET \cup TN$.

**Solution:**

$$
\begin{aligned}
tt(\texttt{SEQ}) &= \{\texttt{DTEp \_dte; SEQp next}\} \\
tt(\texttt{SEQp}) &= \texttt{SEQ*} \\
tt(\texttt{DTE}) &= \{\ \texttt{uint label; DTEp father; DTEp fson; DTEp bro}\ \} \\
tt(\texttt{DTEp}) &= \texttt{DTE*} \\
tt(\texttt{\$gm}) &= \{\ \texttt{DTEp programtree}\ \} \\
tt(\texttt{\$nson}) &= \{\ \texttt{DTEp p; uint j; DTEp result}\ \} \\
tt(\texttt{\$fseq}) &= \{\ \texttt{DTEp seq; SEQp current}\ \}^1 \\
tt(\texttt{\$se}) &= \{\ \texttt{DTEp seq ; uint index}\ \} \\
tt(\texttt{\$main}) &= \ \{\ \}
\end{aligned}
$$

$$
\begin{aligned}
ft(\texttt{nson}).VN &= \{\texttt{p}, \texttt{j}, \texttt{result}\} \\
ft(\texttt{nson}).p &= 2 \\
ft(\texttt{nson}).body &= \texttt{p = p*.fson; ...; return result} \\
ft(\texttt{nson}).t &= \texttt{DTEp} \\
ft(\texttt{fseq}).VN &= \{\texttt{current}, \texttt{seq}\} \\
ft(\texttt{fseq}).p &= 1 \\
ft(\texttt{fseq}).body &= \\
ft(\texttt{fseq}).t &= \texttt{SEQp} \\
ft(\texttt{se}).VN &= \{\texttt{index}, \texttt{seq}\} \\
ft(\texttt{se}).p &= 2 \\
ft(\texttt{se}).body &= \\
ft(\texttt{se}).t &= \texttt{DTEp} \\
ft(\texttt{main}).VN &= \{\} \\
ft(\texttt{main}).p &= 0 \\
ft(\texttt{main}).body &= \ \texttt{...return 0} \\
ft(\texttt{main}).t &= \texttt{uint}
\end{aligned}
$$

**bool** $1, 0$

**char** $\texttt{a}, \texttt{b}$

**int** $-1, -2$

**Saarland University**
**Department 6.2 - Computer Science**
**Prof. Dr. W. J. Paul**
**Jonas Oberhauser**

# System Architecture - SS15

**Exercise Sheet 8(due: June 15, 2015)**

**uint** $1, 2$

**DTEp** $null, 0$

**DTE**

$$u(x) = \begin{cases} 12 & x = \texttt{label} \\ null & \text{o.w.} \end{cases}$$

$$v(x) = \begin{cases} 12 & x = \texttt{label} \\ \texttt{gm.alkohol\_tötet[2]} & x = \texttt{bro} \\ null & \text{o.w.} \end{cases}$$

**SEQp** $null, (\texttt{hamster}, 4)$

**SEQ**

$$u(x) = null$$
$$v(x) = 12$$

## Aufgabe 4: (3)

(a) [1 point] Wieso gibt es Kontextbedingungen?

(b) [2 points] Geben Sie 3 Kontextbedingungen an und erklären Sie ihren Nutzen.

**Solution:** Wir wollen Programme so einschränken, dass wir Bequem Semantik für unsere Programme definieren können. Dazu benuzten wir zuerst eine Kontextfreie Grammatik, die den Programmtext bereits Stark einschränkt. Allerdings können Kontextfreie Grammatiken nicht alle Einschränkungen für uns übernehmen: zum Beispiel hängt die Korrektheit eines Funktionsaufrufs u.a. von den im Programm deklarierten Funktionen ab, d.h., vom Kontext, in dem sich der Aufruf befindet. Solche Bedingungen nennen wir Kontextbedingungen.

Andere Bedingungen sind z.B., dass die Variablennamen u. Funktionsnamen einzigartig sind (sonst müssten wir Typinferenz machen), oder das die Typen bei Zuweisungen Ãœbereinstimmen (sonst müssten wir uns auf Typcasting einlassen).