ulm university universität
uulm

Labor Softwareentwurf mit Multiparadigmen-Programmiersprachen
H. Falk, A. Luppold

Versuch 6
vom 13. Januar 2014

Bearbeitung bis: 26. Januar 2014

Generische Programmierung II

Versuch 6-1: Theorie (keine Punkte)

Entfällt.

Versuch 6-2: Listenprogrammierung zur Compilezeit (20 Punkte)

a) Programmierenm Sie eine Liste, die zur Compilezeit erzeugt wird. Eine Liste wird hierzu rekursiv als entweder die leere Liste, oder ein Listenelement, dessen Nachfolger eine Liste ist, definiert. Setzen Sie nun diese Definition durch zwei geeignete Typen (gegebenenfalls als Templates) um. Eine Liste soll folgendermaßen definiert werden können:

```
1 typedef ListElement<1, ListElement<2, ListElement<3, ListElement<4, EmptyList>>>> list;
```

b) Erstellen Sie eine Template-Klasse, die zur Compilezeit die Summe aller Listenelemente berechent.

Versuch 6-3: Überladen von Operatoren (10 Punkte)

Überladen Sie den Operator << für eine Instanz einer Liste so, dass der folgende Code (zur Laufzeit) die Liste auf der Standardausgabe ausgibt:

```
1 typedef ListElement<1, ListElement<2, ListElement<3, ListElement<4, EmptyList>>>> list;
2 list l;
3 std::cout << l << std::endl;
```

Hinweis: Der Operator muss hierfür für ein Listenelement und für die lere Liste überladen werden.

Versuch 6-4: Arbeiten mit Templates (10 Punkte)

a) Erstellen Sie eine Template-Klasse, die zwei Ihrer Listen konkateniert.

b) Erstellen Sie eine Template-Klasse, die die Reihenfolge der Listenelemente umkehrt. Hinweis: Sie können hierzu den Programmcode aus der vorherigen Teilaufgabe beliebig wiederverwenden.