



**BERGISCHE  
UNIVERSITÄT  
WUPPERTAL**

Prof. Dr. Hans-Jürgen Buhl  
Praktische Informatik/Numerik

Fachbereich C  
Mathematik und Naturwissenschaften,  
Mathematik und Informatik

E-MAIL buhl@math.uni-wuppertal.de

WWW www.math.uni-wuppertal.de/~buhl

DATUM 17. Dezember 2013

## **generische Programmierung**

WS 2013/2014 – Übungsblatt 9

Ausgabe: 17. Dezember 2013

Abgabe bis 8. Januar 2014 an: [kheidsch@studs.math.uni-wuppertal.de](mailto:kheidsch@studs.math.uni-wuppertal.de)

### **Aufgabe 1. Compilezeit Fakultät**

Die Template-Funktion `fact` „berechnet“ zur Compilezeit die Fakultät:

```
// factorial as an integral_constant
#include <iostream>
#include <type_traits>

template <unsigned n>
struct fact : std::integral_constant<int, n * fact<n-1>::value> {};

template <>
struct fact<0> : std::integral_constant<int, 1> {};
```

(Früher wurde dazu etwa

```
template<int i>
class fact {
public:
    static const long long result = i * fact<i-1>::result;
};
```

```
template<> class fact<1> {
public:
    static const long long result = 1;
};
```

benutzt.)

Ergänzen Sie diesen Quellcode um ein Test-Hauptprogramm. Wie wird `fact` aufgerufen? Wozu kann es benutzt werden, da es eine Compilezeit-Konstante produziert? Wie überprüft man diese Compilezeit-Evaluierung?

### Aufgabe 2. *Compilezeit Ganzzahlpotenz*

Schreiben Sie eine ähnliche Template-Klasse zur Berechnung von  $n^m$  für ganzzahlige  $n$  und  $m$ . Beschreiben Sie eine Nutzenanwendung für dieses Template, die die Compilezeit-Evaluation deutlich in den Vordergrund stellt.

Wo setzt die C++-Einschränkung an Template nontype-Parameter der Einsatzfähigkeit dieser Technik Grenzen?

### Aufgabe 3. *BCCL Concepts unter der Lupe*

Untersuche

[/boost/concept\\_check.hpp](#)

auf die Konzepte `UnsignedInteger`, `Comparable`, `Container`, `AssociativeContainer`, `UniqueAssociativeContainer` und `SimpleAssociativeContainer` hin.

Erstelle jeweils eine umgangssprachliche Beschreibung der in ihnen enthaltenen Requirements.

Welche Typen werden in `/boost/concept_check.hpp` durch Templatespezialisierung als `UnsignedInteger` modellierend gekennzeichnet? Was dient in der BCCL also als (nichtabbildender) Ersatz für die `concept_maps` von `ConceptC++`?

### Aufgabe 4. *Concepts der SGI STL*

Erstellen Sie eine Liste aller Konzepte in

[SGI STL](#)

und skizzieren Sie deren Verfeinerungshierarchie.

### Aufgabe 5. *OOP Design-Pattern Observer*

Lesen Sie

[http://de.wikibooks.org/wiki/Muster:\\_Observer](http://de.wikibooks.org/wiki/Muster:_Observer)

und

<http://www.vincehuston.org/dp/observer.html>

Wie unterscheiden sich die beiden Artikel voneinander? Beschreiben sie dasselbe Entwurfsmuster?

Welche Typen von Entwurfsmustern der OOP gibt es nach

<http://de.wikibooks.org/wiki/Muster?>