



Generische Programmierung (Spezielle Kapitel der praktischen Informatik)

WS 2012/2013 – Übungsblatt 9

20. Dezember 2012

Abgabe: bis 10. Januar 2013 an

Fazin.Ghorban@studs.math.uni-wuppertal.de

Aufgabe 1. *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 Templatedespezialisierung als `UnsignedInteger` modellierend gekennzeichnet? Was dient in der BCCL also als (nichtabbildender) Ersatz für die `concept_maps` von `ConceptC++`?

Aufgabe 2. *Concepts der SGI STL*

Erstellen Sie eine Liste aller Konzepte in

[SGI STL](#)

und skizzieren Sie deren Verfeinerungshierarchie.

Aufgabe 3. Lesen Sie das 2008 auf Konzepte hin geänderte Kapitel 26

[Draft N2736](#)

des neuen C++-Standarts.

Welche Konzepte wurden in `numeric_concepts` bereitgestellt? Wo wird `ArithmeticLike`, wo `FloatingPointType` benutzt? Warum?

Wie sehen die Requirements von `accumulate()` in 26.6, wie diejenigen von `inner_product()` aus?

Aufgabe 4. *Template binary*

Testen Sie:

```
template <unsigned long N>
struct binary
{
    static unsigned const value
        = binary<N/10>::value * 2 // prepend higher bits
        + N%10;                  // to lowest bit
};

template <> // specialization
struct binary<0> // terminates recursion
{
    static unsigned const value = 0;
};
```

Was wird hier berechnet? Lassen Sie eine Tabelle berechneter Werte ausdrucken. Welche ähnlichen Anwendungen von Templates erscheinen Ihnen nützlich?