



**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 11. Dezember 2014

generische Programmierung

WS 2014/2015 – Übungsblatt 8

Ausgabe: 15. Dezember 2014

Abgabe bis 5. Januar 2015 an: <mailto:125319@uni-wuppertal.de>

Aufgabe 1. *Template-Funktion mit array-Parametern*

Vergleichen Sie

```
#include <iostream>
```

```
#include <numeric>
```

```
template <typename Type>
```

```
Type sum(Type *tp, size_t n)
```

```
{
```

```
    return std::accumulate(tp, tp+n, Type());
```

```
}
```

```
int main()
```

```
{
```

```
    int x[10];
```

```
    for(int l = 0; l < 10; l++)
```

```
        x[l] = l;
```

```
    std::cout << sum(x, 10) << std::endl;
```

```
    return(0);
```

```
}
```

mit

```
#include <iostream>
```

```
#include <numeric>
```

```
template <typename Type, size_t n>
```

```
Type sum(const Type (&tp)[n])
```

```
{
```

```

    return std::accumulate(tp, tp+n, Type());
}

int main()
{
    int x[10];
    for(int i = 0; i < 10; i++)
        x[i] = i;

    std::cout << sum(x) << std::endl;

    return (0);
}

```

Welche Vor- und Nachteile bietet die erste Variante gegenüber der zweiten? Schreiben Sie eine Template-Funktion `unsigned int get_size(.)`, die die Anzahl der Komponenten eines übergebenen Arrays als Funktionsergebnis liefert.

Aufgabe 2. *InputIterator*

Beschreiben Sie umgangssprachlich in eigenen Worten die einzelnen Requirements des Konzepts `InputIterator` in [Iterator-Concepts](#).

Welche Konzepte müssen die dem Iterator assoziierten Typen modellieren? Skizzieren Sie die Verfeinerungshierarchie aller Iteratoren.

Aufgabe 3. *Draft Proposal: Dynamic Libraries in C++*

Wie sollten gemäß

<http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2003/n1496.html>

shared Objekte (so's) in C++ systemunabhängig standardisiert werden?

Warum wird dieser Erweiterungsvorschlag nach

<http://herbsutter.wordpress.com/2007/02/07/iso-c0x-complete-public-review-draft-in-october-2007/>

nicht in C++0x realisiert werden?

Aufgabe 4. *Draft Proposal: Modules in C++*

Welche Vorteile wären mit Modulen in C++ verbunden:

<http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2006/n2073.pdf>

Wie soll die Syntax aussehen? Welche Vorteile brächten Module gegenüber der momentanen Praxis (welche ist das)?

Aus welchen Gründen wurden Module noch nicht in C++11 aufgenommen (vgl.

<http://herbsutter.wordpress.com/2007/02/07/iso-c0x-complete-public-review-draft-in-october-2007/>)?

Aufgabe 5. *lambda-Ausdrücke/Funktionslitterale/anonyme Funktionen*

Lesen Sie den Abschnitt *Lambda Expressions* in

[Lambda Expressions](#)

sowie die Artikel

[Lambda functions \(seit C++11\)](#)

und

[Lambdas](#).

Wie werden anonyme Funktionen etwa beim Schleifen-Entrollen eingesetzt. Was unterscheidet sie von Funktionsobjekten ([Function objects](#) und [Function Objects \("Functors"\) in C++](#))?