



**BERGISCHE  
UNIVERSITÄT  
WUPPERTAL**

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

Fakultät für  
Mathematik und Naturwissenschaften,  
Mathematik und Informatik

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

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

DATUM 25. Januar 2017

## **generische Programmierung**

WS 2016/2017 – Übungsblatt 11

Ausgabe: 26. Januar 2017

Abgabe bis 2. Februar 2017 an: <mailto:gregor.hildebrand@uni-wuppertal.de>

### **Aufgabe 1. *p*-Norm**

Erstellen Sie einen generischen Algorithmus

```
template <int p = 2, typename InputIter, typename T>  
T pNorm(InputIter first, InputIter last, T init)
```

(vgl. [http://de.wikipedia.org/wiki/Normierter\\_Raum](http://de.wikipedia.org/wiki/Normierter_Raum)).

### **Aufgabe 2. *N*-Queens**

Lesen Sie

<http://accu.org/index.php/journals/424>

und referieren Sie die Anwendung der Metaprogrammierung für das N-Queens-Problem.

Testen Sie das Programm selbst aus. Was ist das Hauptproblem des Einsatzes von Compiletime-Metaprogrammen?

### **Aufgabe 3. *Vorbedingungen in Templates***

Demonstrieren Sie die Verfahrensweisen von Abschnitt 2.7, um die Template-Metafunktion `fact` (Aufgabe 1 von Übungsblatt 9) vor dem Aufruf mit einem negativen Templateparameter-Wert zu schützen. Testen Sie!

Warum nennt man `fact` in diesem Zusammenhang eine Metafunktion und nicht einfach eine Funktion?

### **Aufgabe 4. *Vorbedingungen in Templates: Fortsetzung***

Demonstrieren Sie analog, wie Sie die Template-Metafunktion `template<int n, int m> struct cpower::result` gegen unsinnige Parameter absichern können.

### **Aufgabe 5. *Vorbedingungen in Templates: Fortsetzung 2***

Demonstrieren Sie analog, wie Sie die Template-Metafunktion `template <unsigned long N> struct binary::value` gegen einen unsinnigen Parameter absichern.