



## Programmieren 3 AI – Praktikum

# Übungsblatt 8

1 Punkt

Sebastian Flothow

2011-12-05

**Abnahme:** 2011-12-12

Zur Abnahme ist die persönliche Anwesenheit erforderlich. Bearbeiten Sie die Aufgaben so, dass Sie nicht nur funktionierenden Code vorzuweisen haben, sondern diesen auch erläutern und Fragen dazu beantworten können. Zusätzlich sind die Lösungen am Tag der Abnahme per Email an [sebastian@flothow.de](mailto:sebastian@flothow.de) einzusenden; dies dient u.a. der Prüfung auf eventuelle Plagiate.

Es muss für jede Aufgabe eine getrennte Lösung in einem separaten Verzeichnis vorliegen. Statten Sie jedes Abgabeverzeichnis mit einem Makefile aus. Dieses muss ein Default-Target haben, dass Ihr Programm mit `g++ -Wall -Wextra -pedantic` kompiliert und linkt, sowie das Target „`clean`“, dass die erzeugten Dateien wieder entfernt. Rufen Sie vor dem Packen `make clean` auf (d.h. die eingesandten Archive sollen nur Quelltexte enthalten, keine Kompilate).

Senden Sie Ihre Abgabe als `tar.gz`-Archiv(e). Diese müssen so gepackt sein, dass jedes Archiv genau ein Verzeichnis enthält, dessen Name (bis auf das Suffix) mit dem des Archivs übereinstimmt; ein solches Archiv kann durch einen Aufruf der Form `tar -czf verzeichnis.tar.gz verzeichnis` erzeugt werden.

Beim Kompilieren der Programme mit `g++ -Wall -Wextra -pedantic` dürfen keine Fehler oder Warnungen auftreten. Stellen Sie sicher dass Ihr Programm keine Speicherlecks aufweist und nicht auf nichtallozierten oder uninitialisierten Speicher zugreift; dies ist bei der Abnahme mit Valgrind unter Beweis zu stellen.

### Aufgabe 1

Schreiben Sie ein Programm, dass eine Folge von Gleitkommawerten (`double`) einliest bis End of File erreicht ist und anschließend die eingelesene Folge aufsteigend sortiert und

wieder ausgibt. Das Programm soll außer der `main` keine weiteren Funktionen enthalten. Programmieren Sie keine Schleifen, sondern verwenden Sie ausschließlich Algorithmen der Standardbibliothek. Sie können davon ausgehen dass der Benutzer korrekte Eingaben macht, d.h. eine Fehlerprüfung ist nicht nötig.

## **Aufgabe 2**

Übernehmen Sie die Klasse `Rational` von Blatt 6 Aufgabe 2/3. Ergänzen Sie die Klasse so, dass wenn ein Bruch mit dem Nenner 0 entstände, stattdessen eine Exception geworfen wird. Verwenden Sie dafür eine geeignete Exception-Klasse aus der Standardbibliothek, und versehen Sie die Exceptions mit einer aussagekräftigen Beschreibung der Ursache.

Schreiben Sie in der `main` geeigneten Testcode, der alle Fehlerfälle durchspielt. Die Exceptions sollen jeweils gefangen und mit ihrer Beschreibung ausgegeben werden.