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. Juni 2015

Softwarequalität

SS 2015 – Übungsblatt 9
Ausgabe: 17. Juni 2015
Abgabe bis 24. Juni 2015 an: *mailto:125319@uni-wuppertal.de*

Aufgabe 1. *Umgangssprachliche Spezifikation*

Zeigen Sie, dass das Parkplatzproblem genau eine Lösung der in der Vorlesung spezifizierten Art für alle Eingabegrößen besitzt, die die Vorbedingungen erfüllen. Warum sollte der Auftraggeber der Software `Parkplatzproblem` mit den Vorbedingungen einverstanden sein? Wie könnte eine Softwarelösung aussehen, die ohne den obigen Beweis die Einhaltung des Contracts für sich in Anspruch nimmt?

Aufgabe 2. *relative/absolute Abweichung*

Beim Arbeiten mit Gleitkommazahlen ist die Benutzung der relativen Abweichung empfehlenswert. Warum? (Lesen Sie dazu die in der Materialsammlung zitierten Quellen.)

Wie ist die Maschinengenauigkeit ε für z.B. den Datentyp `double` definiert? Und wie für den Datentyp `long double`? (Überprüfen Sie die Werte in einem kurzen C++-Programm.)

In der Technik ist die alleinige Benutzung der relativen Abweichung als Kleinheitskriterium nicht immer geeignet. Warum? Was sollte gemäß

http://realtimecollisiondetection.net/blog/?p=89

als Alternative benutzt werden? Wie ist dazu die Hilfsfunktion `approximatelyEqualTo()` abzuändern?

Aufgabe 3. *Vervollständigung der Wuerfel-Contracts*

Die Contracts der Klasse Wuerfel verbieten den Methoden `Oberflaeche()`, ... nicht, den Wert des Attributs `Laenge` zu ändern.

Ergänzen Sie die Contracts der entsprechenden Methoden gemäß dem Beispiel der Materialsammlung, um die entsprechende Frame-Bedingung (Benutzung von `ID()` und `ENSURE()`) und testen Sie.

Aufgabe 4. *const-Methoden*

Ändern Sie die Klasse `Wuerfel` des letzten Übungsblatts alternativ zu Aufgabe 3 dadurch ab, dass Sie die abgeleiteten Abfragen zu `const`-Methoden machen und testen Sie wiederum.

Warum ist diese Vorgehensweise sinnvoller? Warum sollte auch in `const`-Methoden die Gültigkeit der Klassen-Invariante bei Methodenbeginn überprüft werden?

Aufgabe 5. *const correctness*

Was versteht man gemäß

http://www.cs.uregina.ca/Links/class-info/210/C++FAQ/const-correctness.html

unter `const correctness`? Was sagt http://en.wikipedia.org/wiki/Const-correctness?

Was sind konstante Klassen-Methoden?