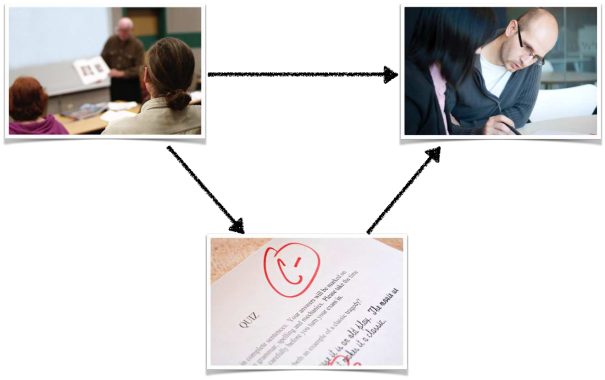
Tutorenschulung Informatik
Kapitel 4: Übungsaufgaben

Jan Vahrenhold

Institut für Informatik
Westfälische Wilhelms-Universität Münster

Wintersemester 2018/2019



Auch Übungen können im Sinne des „alignment“ gestaltet werden.

Notwendigkeit:

- Transparenz der Leistungserwartungen.
- Vergleichbarkeit der Leistungserwartungen in verschiedenen Tutorien.

Eigenschaften von Bewertungsschemata:

- Vorgabe von Bewertungsschwerpunkten.
- Sicherstellung der Vergleichbarkeit der Korrekturen.
- Idealerweise: Ermessensspielraum für Detailentscheidungen.

Indikatoren für Transparenz der Leistungserwartungen:
- Die Leistungsrückmeldungen erfolgen **zügig und differenziert**.
- [Der Lehrer] **erläutert** seine Leistungsrückmeldungen in klaren, insbesondere für die leistungsschwächeren Schüler nachvollziehbaren Worten.

[Meyer, 2007, S. 117]

Grundgedanken:

- Die Korrektur soll das Erreichen der intendierten Lernziele bewerten.
  - Hierzu notwendig: Verständnis der Lernziele durch die Korrektoren (vgl. Kapitel 3).
- Die Korrektur soll Feedback für die Studierenden darstellen.
  - Hierzu notwendig: Differenzierte Rückmeldung (nicht: „Falsch! 0 Punkte“).
- Die Korrektur soll nicht die Arbeit der Studierenden ersetzen.
  - Konkret: Angabe eines Gegenbeispiel, nicht aber Angabe der korrekten Lösung.
- Die Korrektur soll Feedback für die Lehrenden darstellen.
  - Hierzu notwendig: Identifikation häufig gemachter Fehler.

Bemerkung:

- Es gibt keine absolute Wahrheit bzgl. Bewertungsschemata, da diese immer die Schwerpunktsetzung der Lehrenden bzw. Korrigierenden abbilden.
- Dies gilt insbesondere für die folgenden (eigenen) Beispiele.

Aufgabe 6: (5 Punkte) Realisieren Sie den ADT `Queue` unter Verwendung eines zirkulären Felds. Die Größe des Felds soll bei der Konstruktion als Parameter übergeben werden und sich nicht mehr ändern. Dies bedeutet insbesondere, dass—wie auf Folie 1.28 angemerkt—diese Realisierung nicht mehr vollständig der Spezifikation entspricht; dies soll im Rahmen dieser Aufgabe jedoch kein Problem darstellen.

Verwenden Sie die im Learnweb bereit gestellte Schnittstelle `ADTQueue<T>`.

Kommentieren Sie Ihre Implementierung mit Javadoc und geben Sie JUnit-Testfälle zum Testen Ihrer Implementierung bzgl. der Semantik des ADT `Queue` und der Nebenbedingung bzgl. der maximalen Länge an.

Bewertungsschema:

- Für das Erstellen und Testen jeder der folgenden Komponenten ist jeweils ein Punkt vorgesehen:
  - Konstruktor.
  - Methode `isEmpty`.
  - Methode `enqueue`.
  - Methode `dequeue`.
  - Methode `front`.

  Punktabzüge liegen hier im Ermessen der Korrektorin bzw. des Korrektors.
- Die Tests für den Konstruktor und alle Methoden mit Ausnahme von `enqueue` ergeben sich aus Folie 1.13. Der Test für `enqueue` ergibt sich implizit(!) aus der Aufgabenstellung.
- Kommentare:
  - Die Verwendung von `@Override` ist erwünscht, wird jedoch nicht erzwungen.
  - Wird nicht kommentiert, wird ein Punkt abgezogen.
  - Wird nicht gemäß Javadoc kommentiert, wird ein halber Punkt abgezogen.

Aufgabe 10: (2+2=4 Punkte)

(a) Beweisen Sie formal, dass für $f : \mathbb{N} \to \mathbb{R}^{>0}, n \mapsto (n+1)^2$ die Eigenschaft $f \in \mathcal{O}(n^2)$ gilt.

(b) Betrachten Sie die nachfolgende Funktion:

$$f : \mathbb{N} \to \mathbb{R}^{>0}, n \mapsto \begin{cases} n^2 + 15, & \text{falls } n \text{ gerade,} \\ 16n, & \text{sonst.} \end{cases}$$

Beweisen Sie formal, dass $f \in \mathcal{O}(n^2)$ gilt.

Hinweise:

- Für einen formalen Beweis ist die Angabe der Konstanten c und n_0 aus Definition 2.1 notwendig.
- „Beweise“ der Form „offensichtlich“, „aus der Schule bekannt“, „nur der signifikante Term ist von Bedeutung“ stellen keine im Sinne der Aufgabenstellung gültige Lösung dar.

Bewertungsschema:

(a) In dieser Teilaufgabe sind zwei Punkte zu erreichen.
- Für den (erläuterten) Ansatz, die Nullstellen zu betrachten, wird ein Punkt vergeben.
- Für die korrekte Bestimmung der größeren der beiden Nullstellen wird ein halber Punkt vergeben.
- Für die Angabe beider Konstanten in einer Formel wird ein halber Punkt vergeben.

(b) In dieser Teilaufgabe sind zwei Punkte zu erreichen.
- Für die Bestimmung der Konstanten, aus denen sich ergibt, ab wann $f(n) \leq n^2 + 15$ gilt, wird ein Punkt vergeben.
- Für die Bestimmung der Konstanten, aus denen sich ergibt, dass dann auch $f \in \mathcal{O}(n^2)$ liegt, wird ein Punkt vergeben.

Werden Formulierung der Art verwendet, die explizit ausgeschlossen sind, werden keine Punkte vergeben.

Punktvergabe:

100%:	Richtige Antwort, korrekter Beweis.
40–90%:	Fehlerhafter Beweis: Verwirrend aufgeschrieben, fehlenden Fälle, nicht genau genug.
30–50%:	Richtige Antwort, Beweisidee (aber kein exakter Beweis).
20%:	Richtige Antwort, aber kein Beweis.
10%:	Versuch, Teile des Beweises aufzuschreiben (z.B. Induktionsanfang).
0%:	Keine Abgabe bzw. Abgabe nicht korrekt.

Punktabzüge:

10%:	Pro einfachem mathematischen Fehler.
10%:	Antwort nicht in der geforderten Form.
10–30%:	Schlechte Darstellung des Beweises.
10–50%:	Nichtbeachten der Aufgabenstellung (z.B. anderer Ansatz als gefordert).
10–50%:	Fehlende Begründung für Beweisschritt (abhängig von Bedeutung/Schwierigkeit).

Nach: J. S. Vitter, `http://www.cs.duke.edu/courses/fall01/cps230/Handouts/homeworknote.pdf`

Definition einer Rubrik [Popham, 1997]:

- Richtlinie zur Bewertung der Qualität studentischer Arbeiten.
- Bestandteile:
  - Kriterien, bzgl. derer bewertet wird. — Kriteriendimension
  - Skalen, bzgl. derer die Qualität einzelner Kriterien gemessen wird. — Qualitätsdimension
  - Vorgehen zur Berechnung eines nummerischen Wertes (holistisch/analytisch).

Beispiel für Skaleneinteilungen [Stegeman *et al.*, 2014]:

- Einteilung für Programmierarbeiten:
  1. „not good“: Es existieren große Probleme, die behoben werden müssen.
  2. „almost there“: Es existieren keine große Probleme mehr, aber mehr Übung ist notwendig.
  3. „very good“: Die Ziele wurden erreicht, es könnte aber noch marginal besser sein.
  4. „excellent“: Die Erwartungen wurden übertroffen.

LEVEL	1	2	3	4
Documentation - is the code well-annotated to ensure rapid understanding?				
names	names appear unreadable, meaningless or misleading	names accurately describe the intent of the code, but can be incomplete, fuzzy, lengthy, misspelled	names accurately describe the intent of the code, and are complete, distinctive, concise, correctly spelled	all names in the program use a consistent vocabulary

N.B.: Jede Zeile entlang der Qualitätsdimension enthält konkrete Beschreibungen.

LEVEL	1	2	3	4
	Algorithms - is each part of the code as simple as possible?			
flow	there is deep nesting; code performs more than one task per line; control structures are customized in a misleading way	flow is complex or contains many exceptions; choice of control structures and libraries is inappropriate	flow is simple and contains few exceptions; choice of control structures and libraries is appropriate	flow prominently features the expected path
expressions	expressions are repeated or contain unnamed constants	expressions are complex; data types are inappropriate	expressions are simple; data types are appropriate	expressions are all essential for control flow

Vorgehen [Stegeman *et al.*, 2014]:

- Markieren der am besten zutreffenden Qualitätsstufe pro Kriterium.
  - „Stufe 2“ impliziert, dass keine der in „Stufe 1“ beschriebenen Aspekte zutreffen.
  - „Stufe 4“ impliziert, dass die in „Stufe 3“ beschriebenen Aspekte ebenfalls zutreffen.
- Entscheidung für Berechnung eines nummerischen Werts:
  - Holistisch: Alle Werte werden aggregiert. Es wird eine Note aus dieser Summe abgeleitet.
  - Analytisch: Es erfolgt eine Gewichtung der einzelnen Kriterien.

Potenzielle Schwachstellen [Popham, 1997]:

- Zu feingranulare Kriterien.
  - Kriterien sind nur auf ein spezielles Problem anwendbar. Eine Generalisierung ist nicht möglich.
  - Studierende können keine (Lern-)Transferleistungen erbringen.
- Zu allgemeine Beschreibungen der Qualitätsstufen.
  - Beispiel: „Studierender zeigt sehr gute/gute/schlechte Beherrschung der notwendigen Techniken“.
  - Es werden keine konkreten Lernanlässe/Verbesserungsmöglichkeiten geboten.

Alternative Auswertungsarten:

- Holistische Berechnung, jedoch Multiplikation mit „Vollständigkeitswert“.
  - Beispiel (J. Miles, Carleton College): `http://cs.carleton.edu/faculty/jadrian/cs252/2014.0/rubric.html`
- Direkte Einordnung in Noten(bereiche).
  - Beispiel (D. Laidlaw, Brown University): `http://cs.brown.edu/courses/cs016/docs/rubric_info.pdf`

Qualitätsmaßstab:

- Tutorienbesuch muss für Studierende nützlicher als Lesen einer Musterlösung sein.
- Tutorienbesuch muss echten Mehrwert für Teilnehmer bieten.
  - Neue Perspektive auf Lernziele und Inhalte.
  - Personalisierte(re) Erläuterung von Inhalten und Aufgaben.
  - Konkrete Rückmeldung (auch: positiv!) zu Lernstand bzw. -fortschritt.
  - Einschätzung von Bearbeitungsaufwand für Aufgaben.
  - Gemeinsames Entwickeln von Lernstrategien.

Reaktion auf fehlende oder falsche Abgaben:

- Nicht: Reines Anschreiben einer Beispiellösung!
- Stattdessen: Ausrollen von Lerngelegenheiten.
  - Diskussion: Wie gehe ich an diese Art von Aufgaben heran?
  - Gemeinsames Entwickeln von Lösungen (→ Kapitel „Gruppenarbeit“).
- Hoher Vorbereitungsaufwand für Lehrende, aber sehr hohes Lernpotenzial für Studierende.

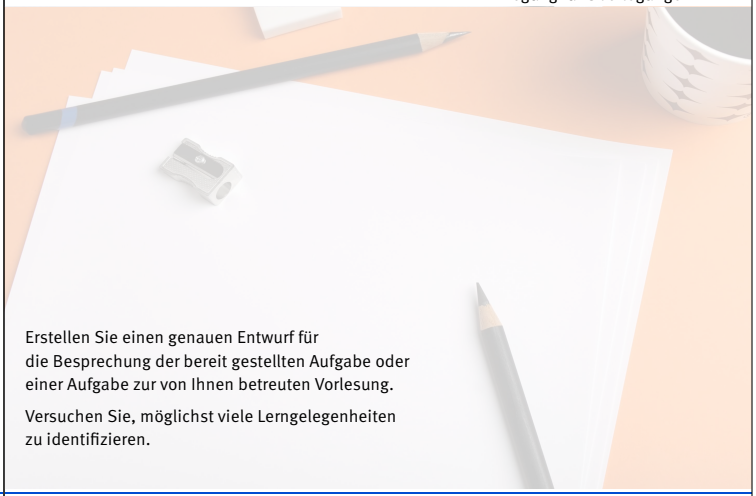
Perspektivenwechsel:

- Hineinversetzen in die Situation von Studierenden.
  - Woran könnte es liegen, dass keine Abgaben erfolgt sind?
  - Welche (früheren) Lernziele wurden anscheinend noch nicht erreicht?
- Hineinversetzen in die Situation von Lehrenden.
  - Welche Schwerpunkte werden bei der Bearbeitung der Lösung gesetzt?
  - Welche Bausteine sind für die Lösung notwendig?
  - Wie können die wesentlichen(!) zur Lösung notwendigen Ideen den Studierenden transportiert werden?

Gestaltungsspielräume:

- Darstellung des „großen Ganzen“ oder von Details?
- Vergleichende Besprechung mehrerer Ansätze?
- Diskussion ausgehend von eigener(?), bewusst falsch konstruierter Lösung?
- Hinausgehen über Beispiellösung?

Diese Fragen sollte man sich bei jeder(!) Besprechung einer Aufgabe stellen!



Erstellen Sie einen genauen Entwurf für die Besprechung der bereit gestellten Aufgabe oder einer Aufgabe zur von Ihnen betreuten Vorlesung.

Versuchen Sie, möglichst viele Lerngelegenheiten zu identifizieren.

Korrekturen:

- Nutzen Sie Beispiellösungen und Bewertungsschemata.
  - Entwickeln Sie diese ggf. mit anderen Tutorinnen und Tutoren.
- Notieren Sie sich, wofür Sie Punkte abgezogen haben.
  - Dies führt zu einer fairen und transparenten Bewertung.
- Geben Sie keine volle Punktzahl, wenn die Lösung Fehler enthält.
  - Lösungen mit voller Punktzahl werden erfahrungsgemäß nicht mehr gelesen.
- Geben Sie hilfreiche (nicht: demotivierende) Rückmeldungen.

Besprechungen:

- Lassen Sie immer zunächst den allgemeinen Lösungsansatz erklären.
  - Gleiches gilt, wenn Sie selbst Lösungen vorstellen.
- Achten Sie darauf, dass vorgestellte Lösungen korrekt und vollständig sind.
  - Die Lösung muss von denen verstanden werden, die die Aufgabe nicht gelöst haben.
- Lassen Sie keine fehlerhaften Aussagen an der Tafel stehen.

Umgang mit eigenen Fehlern:

- Wenn Sie selbst einen Fehler gemacht haben, korrigieren Sie diesen sofort.
  - Genügen Sie den Anforderungen, die Sie an Ihre Studierenden stellen.
- Scheuen Sie sich nicht, Wissenslücken einzugestehen und Antworten zu vertagen.
  - Es ist zwingend notwendig, vertagte Antworten unaufgefordert nachzuholen.

Umgang mit Fehlern Anderer:

- Helfen Sie Studierenden, eigene Fehler selbst zu korrigieren.
  - Stellen Sie niemanden bloß, reiten Sie nicht auf Fehlern herum.
- Lassen Sie Fehler guter Studierender nicht durch sehr gute Studierende korrigieren.
  - Die Korrektur durch „bessere“ *peers* ist demotivierend.
- Stellen Sie nicht ohne Grund Vorlesungsinhalte, Übungsaufgaben oder das Verhalten anderer Lehrpersonen in Frage.
  - Tutoren arbeiten gemeinsam mit den Lehrenden daran…

Bewertungsschemata:

- Transparenz der Leistungserwartungen.
- Vergleichbarkeit der Leistungserwartungen in verschiedenen Tutorien.
- Ausrichtung an Lernzielen und Schwerpunkten.

Rubriken:

- Kriterien mit je einer eigenen Skala.
- Beispielhafte Qualitätsmaßstäbe.

Lerngelegenheiten:

- Konstruktiver Umgang mit fehlenden oder falschen Abgaben.

Literaturverzeichnis

[Meyer, 2007] Meyer, Hilbert: *Was ist guter Unterricht?* Cornelsen Scriptor, Berlin, 4. Auflage, 2007.

[Popham, 1997] Popham, W. James: What's wrong—and what's right—with rubrics. *Educational Leadership* 55(2):72–75, Oktober 1997.

[Stegeman *et al.*, 2014] Stegeman, Martijn, Erik Barendsen und Sjaak Smetsers: Towards and empirically validated model for assessment of code quality. In: Simon und Päivi Kinnunen, Hg., *Proceedings of the 14th Koli Calling International Conference on Computing Education Research*, S. 99–108, ACM, New York City, 2014. `http://stgm.nl/quality`.

Bildnachweis

Folie 0:	Mirko Westermeier, 2017
Folie 1:	© `iStockPhoto.com` / vm
Folie 1:	© `iStockPhoto.com` / Janachan
Folie 1:	© `iStockPhoto.com` / Dougall Photography
Folie 2:	© `iStockPhoto.com` / Raffaele Vannucci
Folie 7:	© `iStockPhoto.com` / Gubcio
Folie 10:	© `iStockPhoto.com` / Orchidpoet
Folie 11:	© `iStockPhoto.com` / monkeybusinessimages
Folie 13:	© `iStockPhoto.com` / Rüstem GÜRLER
Folie 14:	© `iStockPhoto.com` / clu
Folie 15:	© `iStockPhoto.com` / clu
Folie 16:	© `iStockPhoto.com` / DNY59