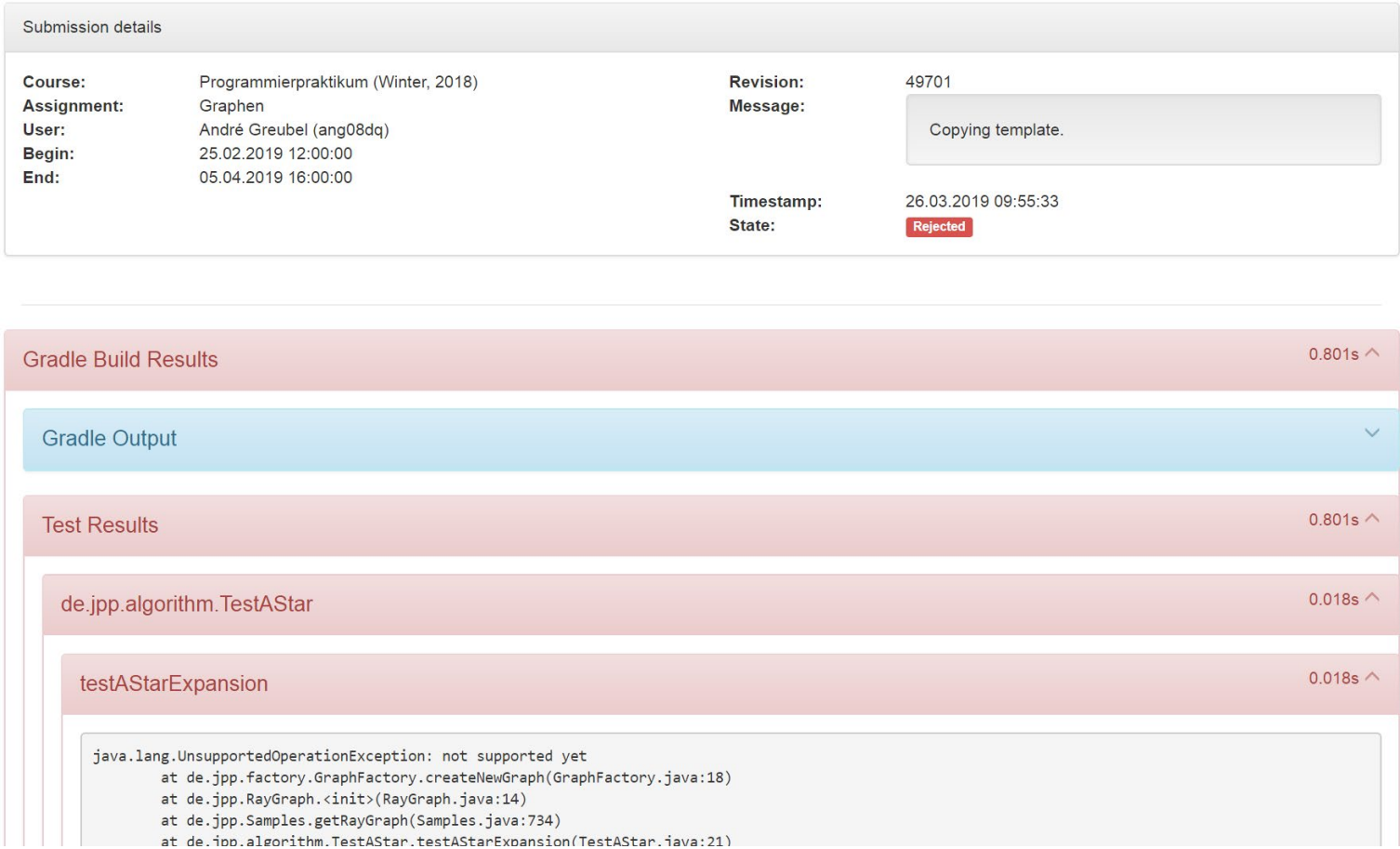
Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Relevanz der Codequalität in einem Praktikum mit automatisch bewerteten Abgaben

André Greubel
Tim Hegemann
Marianus Ifland
Martin Hennecke

Einführung: Blick in die Praxis (PABS)



Zentrale Annahme

Es ist wichtig, dass Programmierer guten Code schreiben können.
Daher sollte das auch im Rahmen der universitären Ausbildung gelehrt werden.

Forschungsfragen

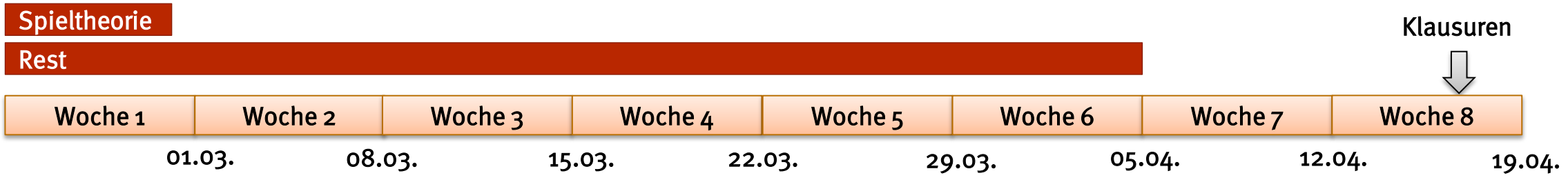
- Wie wichtig ist es Studenten, qualitativ hochwertigen Code zu schreiben?
- Welche automatisch erhebbaren Metriken sind Indikatoren der Codequalität?
- Bestehen Studenten mit gutem Code (gemäß Metriken) das Praktikum häufiger?
- Welche Einflussfaktoren auf das Ergebnis gibt es?

Organisatorische Rahmenbedingungen (WS18/19)

- Struktur

Aufgabe	WINF	MCS	INFO
Spieltheorie	X	X	X
QR-Code	X		X
Graphen	X	X	X
Adressbuch (inkl. GUI)		X	X
Teilnehmer	44	46	95
Zulassungsquote	47.7%	73.9%	54.7%

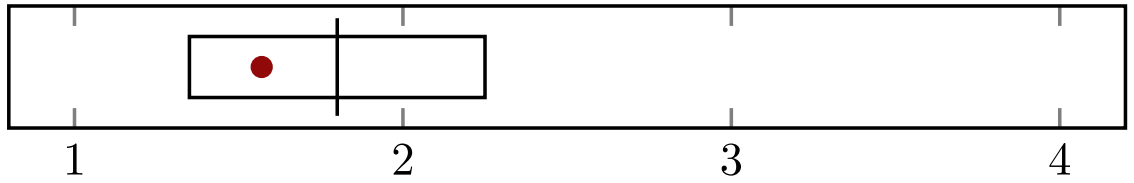
- Ablauf



Erster Schritt: Umfrage

1. Kategorie: Relevanz der Codequalität

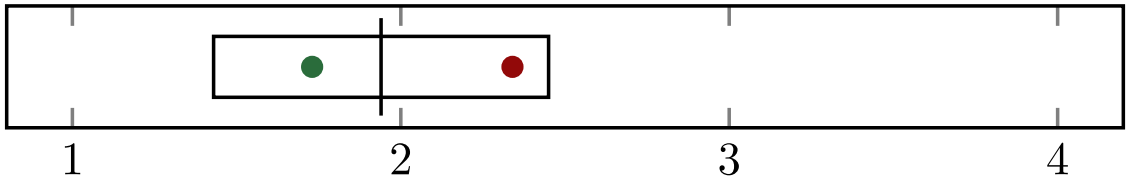
- Die Hauptsache bei der Bearbeitung ist es, die PABS-Test grün zu bekommen



avg: 1.80
stdev: 0.90

WINF: 1.57

- Mir ist es wichtig, dass mein Code über das Bestehen der PABS-Test hinaus inhaltlich richtig ist

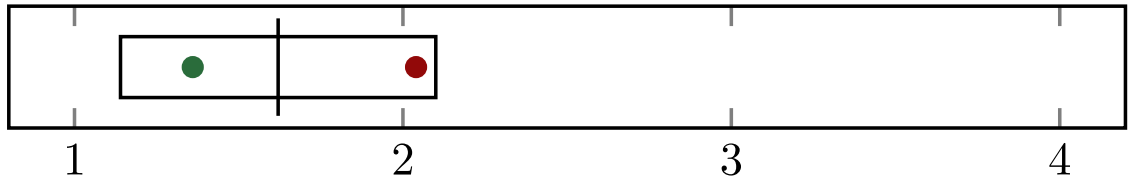


avg: 1.94
stdev: 1.02

INFO: 1.73

WINF: 2.34

- Mir ist es wichtig, dass ich über das Bestehen des Praktikums hinaus lerne gut zu programmieren



avg: 1.62
stdev: 0.96

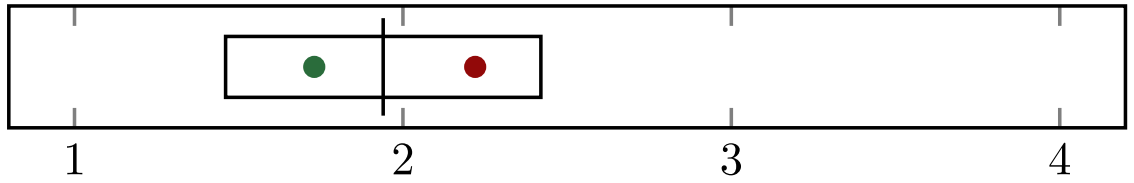
INFO: 1.36

WINF: 2.04

Erster Schritt: Umfrage

1. Kategorie: Relevanz der Codequalität (Fortsetzung)

- Mir ist es wichtig, dass mein Code nicht nur inhaltlich richtig, sondern auch gut lesbar ist.

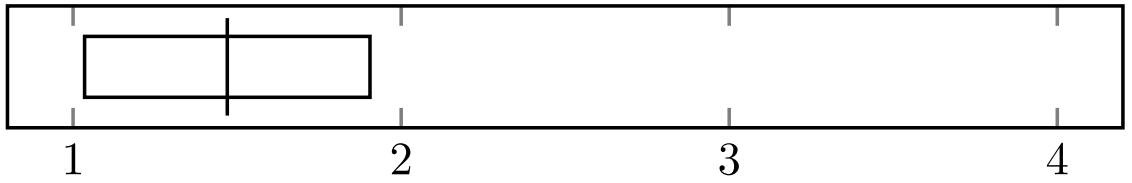


avg: 1.94
stdev: 0.96

INFO: 1.73

WINF: 2.22

- Programmieren ist einfacher, wenn man konsequent darauf achtet, guten Code zu schreiben

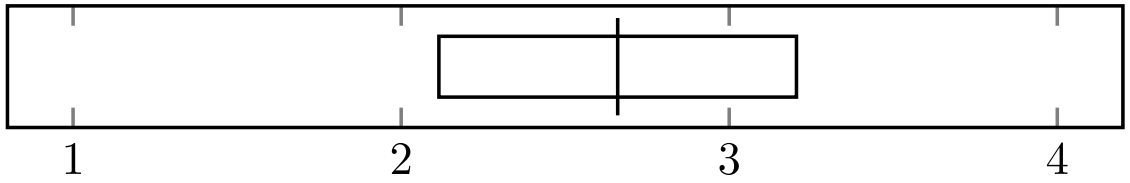


avg: 1.47
stdev: 0.87

Erster Schritt: Umfrage

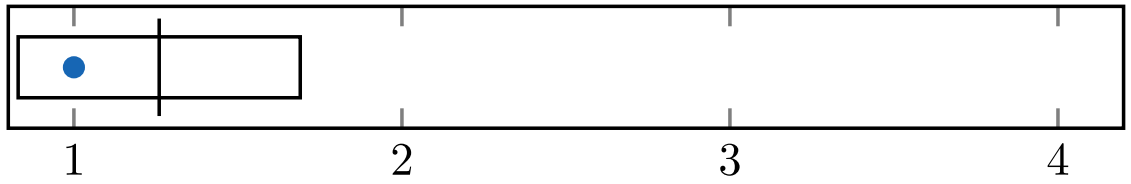
2. Kategorie: Arbeitsweise

- Ich habe meinen eigenen Code refactored (= bereits inhaltlich richtigen Code überarbeitet, um ihn lesbarer zu machen oder besser zu strukturieren)



avg: 2.66
stdev: 1.09

- Ich habe mir Codemetriken ausrechnen lassen, damit ich weiß, wie gut mein Code ist



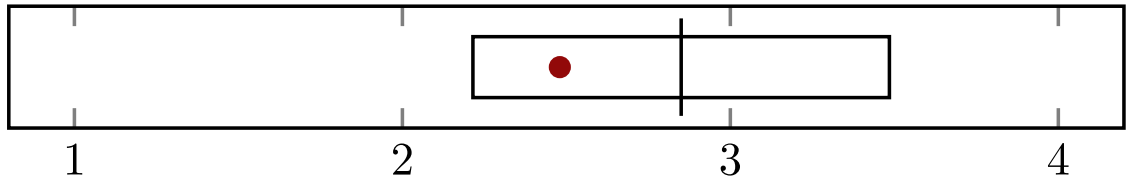
avg: 1.26
stdev: 0.86 MCS: 1.00

Skala hier abweichend: 1: nie ⇔ 4: oft

Erster Schritt: Umfrage

3. Kategorie: Hilfestellungen

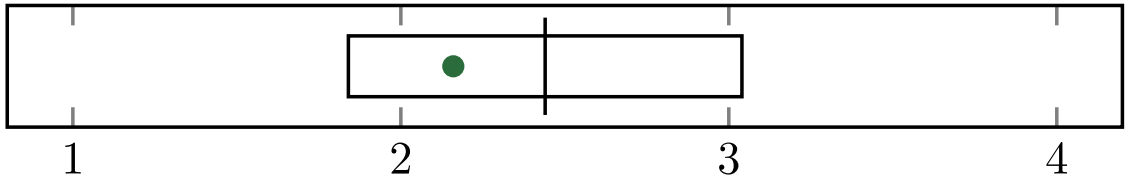
- Die LoC-Angaben in den Klassendiagrammen der Graphen-Aufgabe haben mir bei der Bearbeitung geholfen



avg: 2.85
stdev: 1.27

WINF: 2.48

- In freieren Aufgaben meine eigene Struktur definieren zu können hilft mir, besseren Code zu schreiben



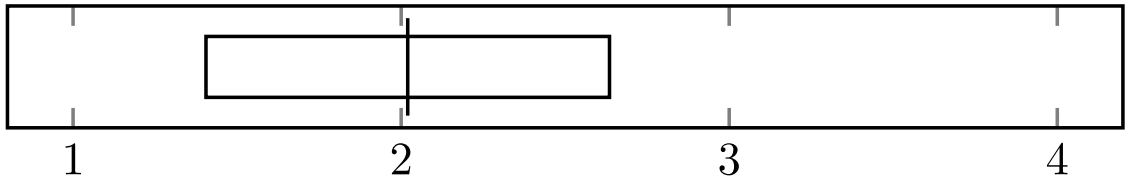
avg: 2.44
stdev: 1.20

INFO: 2.16

Erster Schritt: Umfrage

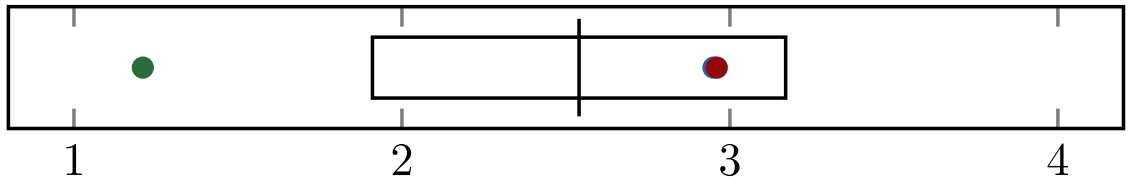
4. Kategorie: Folgerungen

- Ich würde mir mehr Rückmeldung zu meiner Codequalität wünschen



avg: 2.02
stdev: 1.23

- Solange das Praktikum dadurch nicht schwieriger wird sollte auch die Qualität des Codes in die Bewertung mit einbezogen werden (statt nur inhaltliche Korrektheit zu bewerten).



avg: 2.54
stdev: 1.26

INFO: 1.21
MCS: 2.95
WINF: 2.96

Codequalität

- Übliche Codemetriken
  - ↘ Gesamtkomplexität (Weighted Methods per Class, ***WMC***)
  - ↘ Maximale Schachtelungstiefe (***Nesting Depth***)
  - ↘ Maximale Methodenkomplexität (**Most Complex M.**)
- Indikatoren der Strukturiertheit
  - ↘ Lines of Code (***LoC***)
  - ↘ Maximaler Methodenumfang (***Longest Method***)
  - ↗ Anzahl der Methoden (***Methods***)
- Good bzw. Bad Smells
  - ↘ Anzahl der Literale (***Literals***)
  - ↘ Anzahl der Typecasts (***Typecasts***)
  - ↗ Anzahl der Lambdas (***Lambdas***)

Methodische Herangehensweise

1. Erhebung: Zusammenhang Codequalität Einführungsaufgabe ⇔ Bestehen

   - Unterteile Studenten in zwei Gruppen
     1. Gruppe: Einführungsaufgabe bestanden, Klausurzulassung erhalten
     2. Gruppe: Einführungsaufgabe bestanden, Klausurzulassung nicht erhalten

   - Unterschiede in der Metrikverteilung der beiden Gruppen?
     1. Absolute Unterschiede
     2. Wilcoxen-Rangsummentest

Ergebnisse

Metrik-Kürzel	Referenz-Lösung	Stud+ Avg (± StdDev)	Stud- Avg (± StdDev)	Signifikanz (p-Wert)
WMC	66	127.41 (± 12.73)	133.15 (± 17.79)	0.86%
Nesting Depth	2	2.87 (± 0.80)	3.11 (± 0.98)	5.31%
Most Complex M.	5	13.83 (± 3.53)	14.50 (± 3.94)	9.06%
Lines of Code	353	496.64 (± 61.80)	505.56 (± 67.68)	10.84%
Longest Method	12	38.98 (± 15.05)	37.99 (± 9.64)	43.41%
Methods	52	54.22 (± 2.94)	54.83 (± 4.83)	35.14%
Literals	41	87.80 (± 25.20)	93.37 (± 22.58)	0.72%
Typecasts	1	1.86 (± 2.20)	1.97 (± 2.44)	44.40%
Lambdas	14	1.37 (± 3.30)	1.30 (± 3.05)	48.67%

Methodische Herangehensweise

2. Erhebung: Zusammenhang Codequalität Einführungsaufgabe ⇔ Studiengang

- Unterteile Studenten pro Studiengang in zwei Gruppen
  1. Gruppe: Teil des Studiengangs
  2. Gruppe: Anderer Studiengang

- Unterschiede in der Metrikverteilung der beiden Gruppen?
  1. Absolute Unterschiede
  2. Wilcoxen-Rangsummentest

Ergebnisse

Metrik-Kürzel	Durchschnitt (StdDev)	INFO (Signifikanz)	MCS (Signifikanz)	WINFO (Signifikanz)
WMC	129.68 (± 16.34)	127.62 (1.12%)	132.20 (15.76%)	133.48 (4.82%)
Nesting Depth	2.98 (± 0.90)	2.88 (4.28%)	2.89 (34.44%)	3.27 (0.76%)
Most Complex M.	14.08 (± 3.79)	13.56 (0.10%)	14.57 (5.25%)	15.57 (2.26%)
Lines of Code	499.57 (± 65.87)	494.96 (19.61%)	495.22 (19.92%)	516.93 (3.11%)
Longest Method	38.29 (± 12.87)	37.49 (8.29%)	38.98 (28.37%)	41.86 (14.75%)
Methods	54.49 (± 3.95)	54.52 (16.56%)	53.80 (44.00%)	53.89 (16.16%)
Literals	90.08 (± 24.32)	87.82 (20.02%)	90.35 (44.35%)	95.45 (12.83%)
Typecasts	1.92 (± 2.31)	1.64 (7.99%)	2.07 (34.44%)	2.27 (2.20%)
Lambdas	1.43 (± 3.31)	2.03 (0.10%)	0.48 (4.14%)	0.48 (3.14%)
Arbeitszeit	54.06 h	45.12 h	46.94 h	85.25 h

Weitere Ergebnisse

1. Erhebung: Zusammenhang Codequalität Einführungsaufgabe ⇔ Wiederholung
   - Unterteile Studenten in zwei Gruppen
     1. Gruppe: Mindestens eine Abgabe in einem vorherigem Praktikumsdurchlauf
     2. Gruppe: Hat in keinem vorherigen Praktikum eine Abgabe hochgeladen
   - Unterschiedliche zwischen Gruppen in der Metrikverteilung?
     1. Um 11.23% kürzere Methoden (p=0.37%)
     2. Pro Abgabe 0.54 Lambdas weniger (p=1.58%)
   - Relative Position der Wiederholer im alten und neuen Praktikum?
     1. Leiche Verbesserung in allen Metriken
     2. Signifikanz bei Nesting Depth (p=3.51%), Most Complex M. (p=5.80%), Methods (p=3.34%)
     3. Signifikanz auch bei Lambdaanzahl (p=0.13%)

Erste Antworten zu den Forschungsfragen

- Wie wichtig ist es den Studenten, qualitativ hochwertigen Code zu schreiben?
  - Umfrage unter willigen Teilnehmern des Praktikums (88 Antworten, ~40% Rücklauf)
  - Eher wichtig im Praktikum, sehr wichtig außerhalb
- Welche automatisch erhebbaren Metriken sind Indikatoren der Codequalität
  - 9 Metriken erhoben
- Bestehen Studenten mit gutem Code (gemäß Metriken) das Praktikum häufiger?
  - Ja: Gesamtkomplexität (WMC), Einrückungstiefe, Komplexeste Methode, Anzahl Literale
- Welche Einflussfaktoren auf das Ergebnis gibt es?
  - Erfahrung (extrem): Langjähriger Hiwi -> Wesentlich besserer Code, geringere Arbeitszeit
  - Studiengang (deutlich): Informatikaffiner -> Besserer Code, kürzere Arbeitszeit
  - Wiederholer (fast nicht): Nach Durchfallen sind Metriken etwa gleich gut

Ausblick – Weitere Fragen

- Wie Verallgemeinerbar sind die Ergebnisse?
  - Erster Blick auf die Daten: WMC ist ein sehr stabiler Indikator für das Bestehen
  - Weitere Ergebnisse sehr von Aufgaben und Organisationsstruktur abhängig
- Wie gut beschreiben die Metriken die menschliche Auffassung von gutem Code?
  - Wie fein kann ein menschlicher Konsens überhaupt werden (3 Stufen, 6 Noten, 100 Punkte?)
- Welche allgemeinen Kompetenzen und Regeln lassen sich formulieren?
  - Bisher unberücksichtigt: Hintergrundvariable vs. Gerichteter Effekt?
- Wie lässt sich Feedback zur Codequalität automatisch erzeugen?
  - Wäre das didaktisch überhaupt wünschenswert?
  - Reicht die Verarbeitung konkreter Metrikwerte?
  - Wie einfach kann die automatische Erzeugung in die Irre geführt werden?