

# Teaching Programming at Scale

SEUH 2020 – A. Kaplan, J. Keim, Y. R. Schneider, M. Walter, D. Werle, A. Koziolk, R. Reussner

SOFTWARE-ENTWURF UND -QUALITÄT,  
INSTITUT FÜR PROGRAMMSTRUKTUREN UND DATENORGANISATION, KIT-FAKULTÄT FÜR INFORMATIK



# Programmierenlehre - *learning by doing*

## Übungsblatt 01

### Modellieren Sie ein Auto

Lorem ipsum dolor sit amet,  
 consetetur sadipscing elitr, sed  
 diam nonumy eirmod tempor  
 invidunt ut labore et dolore magna  
 aliquyam erat, sed diam voluptua.  
 At vero eos et accusam et justo  
 duo dolores et ea rebum. Stet clita  
 kasd gubergren, no sea takimata  
 sanctus est ...



```

class Car {
  .....
}
  
```



```

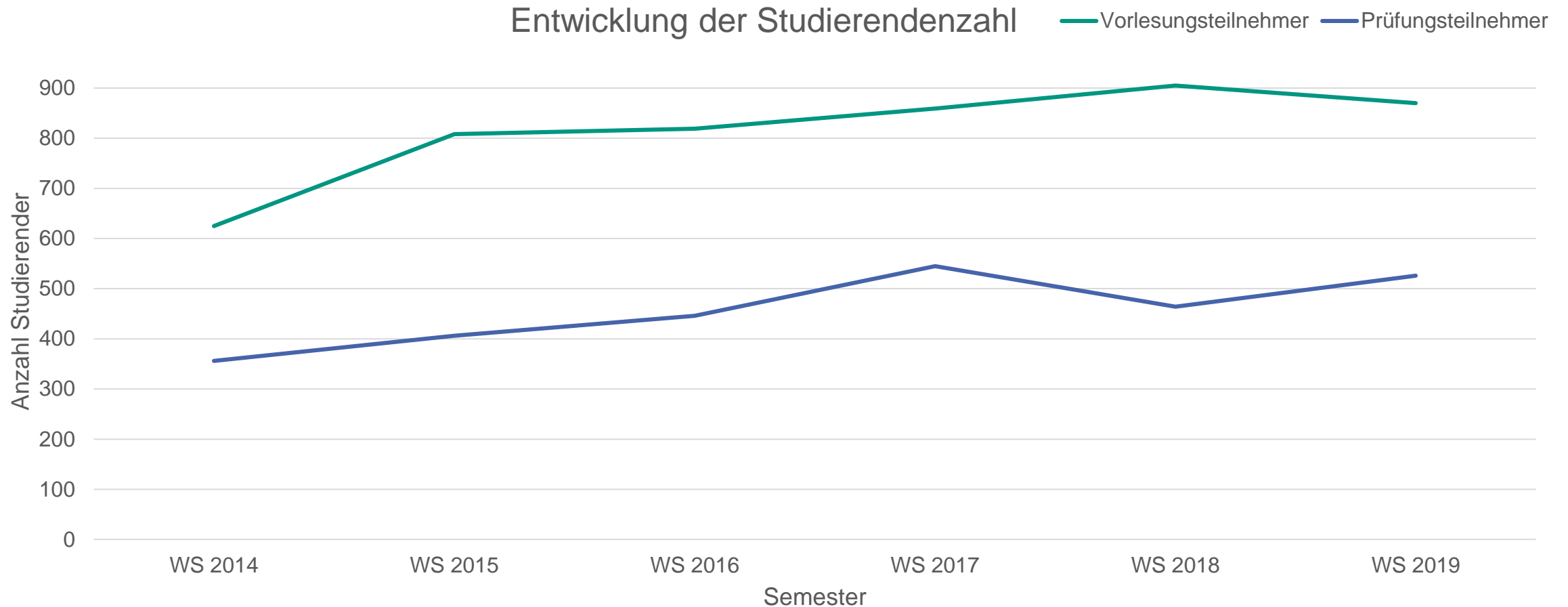
class Vehicle
{
  .....
}
  
```

### Bewertung

- Stil
- Funktionalität



# Steigende Studierendenzahlen



# Steigende Studierendenzahl

Wie bewerten wir effektiv und nachvollziehbar den Stil der Abgaben?

Wie strukturieren wir den Lehrbetrieb gut?

Wie gehen wir mit steigenden Studierendenzahlen um?

Wie organisieren wir die Kommunikation mit den Studierenden möglichst effizient und fair?

Wie stellen wir sicher, dass Studierende Ihre Lösung selbständig erstellen?



# Veranstaltungsaufbau

## Lehrbetrieb:

- Vorlesung – Vorstellung allgemeiner Programmierkonzepte
- Übungsbetrieb - Anwendung der Programmierkonzepte

## Leistungskontrolle:

- 2 umfangreiche Programmieraufgaben mit je ca. 1000 Zeilen Code
- Bewertung von Programmierstil und Funktionalität



# Lehrkonzept – Erfahrungsbericht

## Erfahrung:

- Klare Zeitpläne für Inhalte helfen Studierenden und Lehrenden \*
- Aktivierungsaufgaben während der Vorlesung hilfreich \*
- Implementierung von Spielen beliebt \*

## Offene Punkte:

- Vorstellungsreihenfolge von Algorithmen und Objektorientierung
- Trennung von Tutorieninhalt und Vorlesungsinhalt

\* In Lehrevaluation durch Studierende genannt

# Effektive und verständliche Bewertung

## Herausforderung:

- Manuelles Bewerten von Programmierstil zeitaufwendig
- Korrekturanmerkungen teilweise schwer nachvollziehbar
- Bewertung von Stil teilweise sehr subjektiv

## Lösungsansatz:

- Erweiterung des elektronischen Abgabesystems um Textbausteine
- Vorgegebene Liste der Bewertungskriterien
- Automatische Punkteberechnung auf Basis der Kategorien



# Bewertung – Korrektursicht

Hop  
 GamePlayer  
 Direction

Functionality: 13  
 OO-Modelling: 3

no OOP    static method    visibility  
 getterSetter lists    runtime exceptions  
 code copy inheritance    enum/inheritance  
 object    methods instead of domain  
 bad modelling    static attribute  
 Custom

Comprehensibility: 2

complex code    magic number    IO/UI  
 exception controlflow    try/catch  
 JavaAPI    Java datastructures    Custom

Style: 2

empty JavaDoc    JavaDoc trivial  
 bad identifier    comments  
 code copy methods    instanceof  
 indentation    final  
 utility class: constructor    string references

These are public tests. Remember, we have got a lot more than just these.

## + Example interaction

Success

1 success

- ▶ **Private 01 : passed**
- ▶ **Private 02 : passed**
- ▶ **Private 03 : passed**
- ▶ **Private 04 : passed**
- ▶ **Keep file tests.log : passed**

Main.java	Command.java	Game.java	InvalidParameterException.java	InputException.java	Token.java	Tile.java	
Role.java	Position.java	Player.java	Path.java	Number.java	Hop.java	GamePlayer.java	Direction.java

```

1  /*
2  * Copyright (c) 2019, IPD Reussner. All rights reserved.
3  */
4
5  package edu.kit.informatik.searchformisterx.game;
6
7  import edu.kit.informatik.searchformisterx.data.Direction;
8  import edu.kit.informatik.searchformisterx.data.GamePlayer;
9  import edu.kit.informatik.searchformisterx.data.Hop;
10 import edu.kit.informatik.searchformisterx.data.Path;
11 import edu.kit.informatik.searchformisterx.data.Player;
12 import edu.kit.informatik.searchformisterx.data.Position;
  
```





# Bewertung – Studierendensicht

## Attestation: Final Task 2

for Yves Richard Schneider

### Comment

-----  
Style (-0.5P)

\* -0.5P: javaDoc is empty or nonexistent (Main.java:23).

### Ratings

Style: 1.5

Comprehensibility: 2.0

OO: 3.0

Functionality 13.0

Final grade: 19.5

### Checker results

▼ Private 01 : passed

```

- Place
Tests if placing all stones works correctly.
Terminal
> java edu.kit.informatik.searchformisterx.main.Main
start XIV
OK
place AIR 1 XIV
Error, move not allowed
Line 1: expected 'OK', but got 'Error, move not allowed'
place AIR 3 XIV
  
```

Textbausteine mit Verweis auf Dateiname und Zeile des Punktabzugs

Erreichte Punktzahl je nach Kategorie

Erreichte Gesamtpunktzahl

Ergebnis funktionaler Tests

# Bewertung – Erfahrungsbericht

## Erfahrung:

- Liste mit Stilkorrekturen verringert die Anzahl der Fehlkorrekturen
- Verweise auf Fehler helfen Studierenden die Korrektur nachvollzuziehen
- Semi-automatische Werkzeuge reduzieren die Korrekturdauer

## Offene Punkte:

- Stilkorrektur weiterhin aufwendig
- Aufwendige Wartung und Abhängigkeit von selbstentwickelten Werkzeugen
- Zu allgemeine Fehlerkategorien



# Kommunikation mit Studierenden

## Herausforderung:

- Individuelle Kommunikation mit Studierenden zeitaufwendig
- Für alle Studierende die gleichen Informationen
- Reduzierung von Wiederholungen

## Lösungsansatz:

- Einsatz von Online-Lernplattformen



## Wiki-Funktionalität:

- Dokumentation der Bewertungsgrundlage für Programmierstil
- Einstiegshilfen zur Programmierung mit Java (IDE, Installation Java)

## Foren:

- Inhaltliche und organisatorische Fragen + Ankündigungen
- Aktuelles WS ca. 700 Beiträge
- Hauptsächlich Fragen zum Übungsbetrieb (ca. 85%)



# Kommunikation – Erfahrungsbericht

## Erfahrung:

- Vereinfachung der Kommunikation durch Online-Lernplattformen
- Online-Lernplattformen unterstützen sowohl Lehrende als auch Studierende\*

## Offene Punkte:

- Stärkere Beteiligung von Studierenden beim gegenseitigen Beantworten von Fragen
- Reduzierung von organisatorischen E-Mails

\* In Lehrevaluation durch Studierende genannt

# Zusammenfassung

Wie bewerten wir effektiv und nachvollziehbar den Stil der Abgaben?

- *Automatische Textbausteine*
- *Ausführliche Korrekturrichtlinien*

Wie strukturieren wir den Lehrbetrieb gut?

- *Zeitpläne erstellen*
- *Spiele sind beliebt*

Wie gehen wir mit steigenden Studierendenzahlen um?

Wie organisieren wir die Kommunikation mit den Studierenden möglichst effizient und fair?

- *Nutzen von Online-Lernplattformen*

Wie stellen wir sicher das Studierende Ihre Lösung selbständig erstellen?

- *Nutzen von Plagiaterkennungsprogrammen*
- *Schriftliche Zwischenprüfung*

