

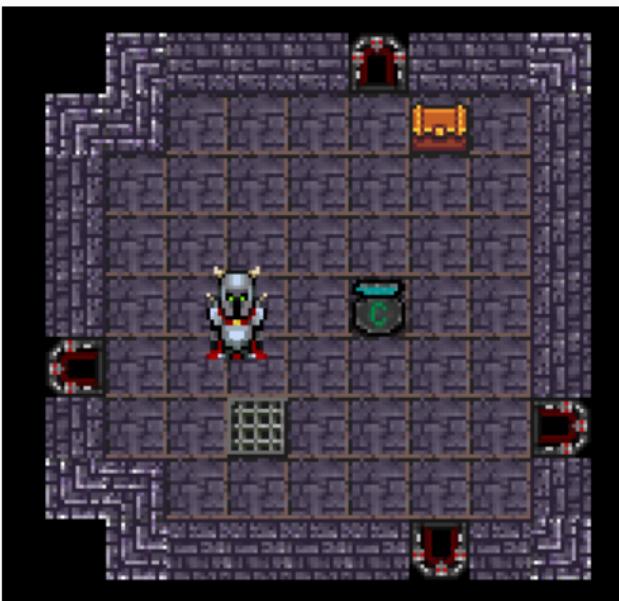
Peer-Feedback, Poster-Sessions und OER in ILIAS-Kursräumen

Finn Amini Kaveh, Carsten Gips (HSBI)

21. November 2023

Unless otherwise noted, this work is licensed under CC BY-SA 4.0.

Lernszenario “Programmiermethoden”: Wir entwickeln ein Spiel



Lernszenario “Programmiermethoden”: Wir entwickeln ein Spiel



Bearbeitung der
Aufgabe

Abgabe der
Lösung im ILIAS

Vorstellung der
Lösung im
Praktikum

Bewertung/
Feedback durch
Lehrende

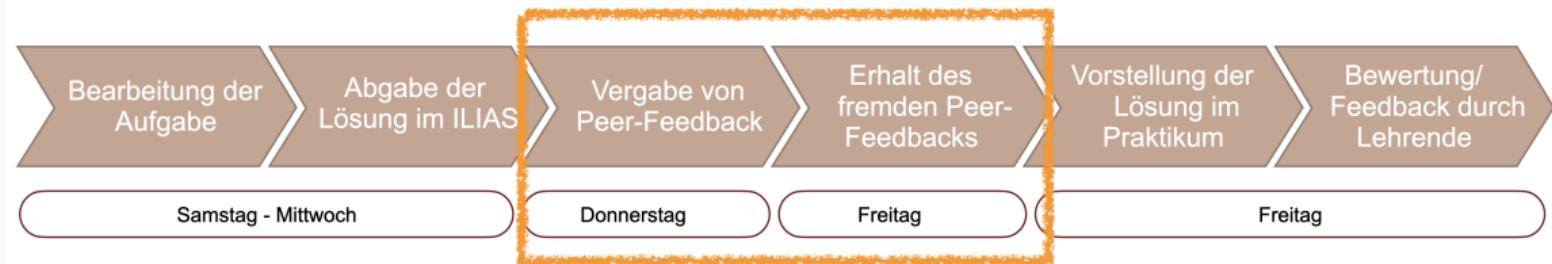
Samstag - Donnerstag

Freitag

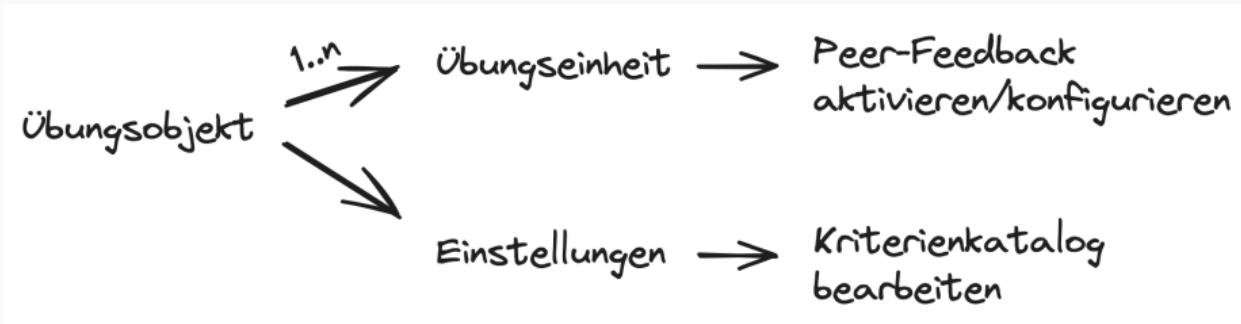
Peer-Feedback zu Übungsaufgaben im ILIAS

Ziele Peer-Feedback: Studierende sollen ...

- Fremde Lösungen (Code) lesen lernen
- Fremde Konzepte bewerten lernen
- Anregungen für ihre eigenen Lösungen bekommen (Spieleentwicklung!)



Peer-Feedback: Kriterienkataloge im Übungsobjekt



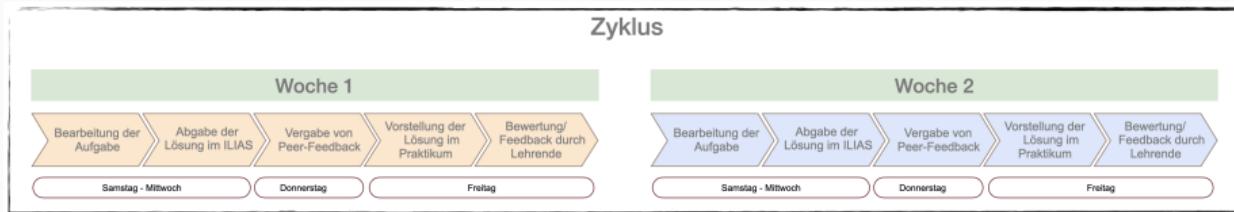
Beispiele für Review-Fragen:

- Wie gut können Sie die Modellierung nachvollziehen? (5-Sterne-Bewertung)
- Was gefällt Ihnen an der Modellierung besonders? (Text)
- Wie könnte die Modellierung verbessert werden? (Text)
- Beurteilen Sie die Dokumentation des Codes und geben Sie Verbesserungshinweise. (Text)
- Kein Review – es handelt sich um die Abgabe meines Teams. (Erfüllt Ja/Nein)

Peer-Feedback: Lessons Learned

- **Fragen:** Lieber Freitext statt Checkboxen bzw. 5-Sterne-Bewertung nutzen
- **ILIAS:**
 - Peer-Feedback lässt sich nur für Einzelabgaben konfigurieren
 - Keine Nachfrist oder individuelle Abgabe möglich
 - Kriterienkataloge lassen sich nicht kopieren
- **Organisation:**
 - Zusätzliche Bearbeitungszeit für Peer-Feedback notwendig
 - Peer-Review = zusätzliche “Abgabe” (aus Studierenden-Perspektive)
 - Höherer Workload für die Studierenden

Mehrwöchige Arbeitszyklen: Orientierung im ILIAS



HSB'I
ILIAS - Das Learning Management System der HSB

Lernaktivität > Campus Minden > Bereich Informatik > Sommersemester 2023 > IFM-2.1 - Programmiermethoden - Gips - SS2023

Aktivitäten bearbeiten

Aktivitäten bearbeiten

Link

Kurs Link

Weiterleiten

Verknüpfung von ILIAS

Wählen Sie ein Ziel für die Verknüpfung des Blocks. Achtung: in diesem Fall darf der Inhalt des Blocks dann nicht mehr weiteren Links enthalten!

Aktiv ab: 12.04.2023 00:00

Aktiv bis: 29.04.2023 00:00

Berechnungszeitraum

Auswählen

Erweitern

Erweiterungen

Achtung!

Falls ein Berechnungszeitraum ausgewählt wird, wird der Block nur Benutzern mit dem gewählten Recht zur Berechnung freigeschaltet.

Suchen > Absuchen

Praktikum: Aktueller Zyklus: Zyklus 1

Wochen 03: Zyklus 1: Konzept

- Konzept: 21.04. - 08:00 Uhr
- Peer-Feedback: 21.04. - 08:00 Uhr
- Praktikum: 21.04.

Wochen 04: Zyklus 1: Implementierung

- Implementierung: 27.04. - 08:00 Uhr
- Peer-Feedback: 28.04. - 08:00 Uhr
- Praktikum: 28.04.

Praktikum: Aktueller Zyklus: Zyklus 2

Wochen 05: Zyklus 2: Konzept

- Konzept: 04.05. - 08:00 Uhr

Wochen 06: Zyklus 2: Implementierung

- Implementierung: 11.05. - 08:00 Uhr

HSB'I
ILIAS - Das Learning Management System der HSB

Lernaktivität > Campus Minden > Bereich Informatik > Sommersemester 2023 > IFM-2.1 - Programmiermethoden - Gips - SS2023

Vorlesung als Mitglied

IFM-2.1 - Programmiermethoden - Gips - SS2023

Achtung Info Mitglieder

Herzlich willkommen zu Programmiermethoden im S23! "Weniger schlecht programmieren" :-)

Alle Stunden finden ausschließlich über Zoom statt:
<https://hsb-i.de/zoom.us/j/190023577?pwd=VmxzZGJtR2MkaHltaS0tOT09>

In Lernmodul finden Sie die Orga, das Skript mit den Lernvideos und die Aufgabenblätter.

Neugkeiten

← 1 2 3 →

Hinweise zur Sonderprüfung
Kurs IFM-2.1 - Programmiermethoden - Gips - SS2023
Datum: 23. Jun 2023, 11:00

Endsemester-Umfrage offensiv
Kurs IFM-2.1 - Programmiermethoden - Gips - SS2023
Datum: 23. Jun 2023, 19:00

Der Test wurde online geschaltet.
Test Programmieren
Datum: 08. Jun 2023, 19:00

Peer-Feedback am OBL eröffnet (ab Freitag)
Kurs IFM-2.1 - Programmiermethoden - Gips - SS2023
Datum: 08. Jun 2023, 17:00

Visions-Workshop: Challenge Überarbeitet
Kurs IFM-2.1 - Programmiermethoden - Gips - SS2023
Datum: 09. Jun 2023, 19:00

Praktikum: Aktueller Zyklus: Zyklus 1

Wochen 03: Zyklus 1: Konzept

- Konzept: 20.04. - 08:00 Uhr
- Peer-Feedback: 21.04. - 08:00 Uhr
- Praktikum: 21.04.

Wochen 04: Zyklus 1: Implementierung

- Implementierung: 27.04. - 08:00 Uhr
- Peer-Feedback: 28.04. - 08:00 Uhr
- Praktikum: 28.04.

Link zu dieser Seite: <https://www.hsbit.de/ilias/> powered by ILIAS (v7.2023-19-20) | Datenschutzerklärung | Impressum

Poster-Galerie im Modul “Künstliche Intelligenz”

Seite einrichten → Spalten-Layout →

Poster im "PNG"-Format
als "Bild/Audio/Video"
einfügen

→ Medienobjekt;
"Vollbild" aktivieren

The screenshot shows a grid of 12 posters from a project titled 'Stundenplanproblem' (Scheduling Problem) using a genetic algorithm. The posters are arranged in three rows and four columns. Each poster contains text, diagrams, and tables related to the scheduling problem.

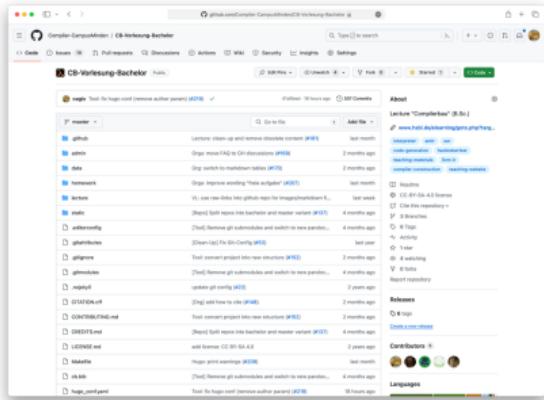
- Row 1:**
 - ZEITPLANUNG MIT DEM GENETISCHEM ALGORITHMUS**: A poster with a large diagram of a genetic algorithm flowchart.
 - STUNDENPLANPROBLEM Constraint-Solving Problem**: A poster with a table of constraints and a flowchart of the problem-solving process.
 - Stundenplanung**: A poster showing a weekly schedule table for a student.
 - Stundenplangenerierung mittels Generischer Algorithmen**: A poster with a flowchart of the planning process and a table of generated schedules.
- Row 2:**
 - EINFÜHRUNG**: A poster with a graph showing the relationship between complexity and time.
 - ZIEL**: A poster with a diagram of a genetic algorithm's search space.
 - METHODE**: A poster with a flowchart of the genetic algorithm's iterative process.
 - ANALYSE**: A poster with a graph showing the analysis of a solution.
- Row 3:**
 - ERGEBNIS**: A poster with a table of results and a note about the final solution.
 - STUDENTENPLANUNG MIT GENETISCHEM ALGORITHMUS**: A poster with a flowchart of the student scheduling process.
 - Das Stundenplan-Problem als CSP**: A poster with a table of constraints and a note about the problem being a Constraint Satisfaction Problem.
 - Das Stundenplan-Problem als CSP**: A second poster with a table of constraints and a note about the problem being a Constraint Satisfaction Problem.

Unterstützung durch Digi-Scouts von DigikoS

Vielen Dank an die Digi-Scouts vom DigikoS-Projekt für die
Unterstützung bei der technischen Umsetzung im ILIAS!

OER-Kurse

Offene Quellen (GitHub)



Export: Lernmodul und Folien

Offener Kursraum (ILIAS)



+

Geschlossener Kursraum (ILIAS)

Beispiel Lernmodul Compilerbau

Lernmodul Startseite

IFM 5.21: COMPILERBAU (WINTER 2023/24)

The diagram shows a flow from C code (`#include <stdio.h>`) through a scanner (represented by three circles) and parser (represented by a grid of nodes) to a code generator (GCC), which then produces assembly code and finally machine code.

Kursbeschreibung

Der Compiler ist das wichtigste Werkzeug in der Informatik. In der Königsdisziplin der Informatik schließen sich der Kreis. Hier kommen die unterschiedlichen Algorithmen und Datenstrukturen und Programmiersprachenkonzepte zur Anwendung.

In diesem Modul geht es um ein grundlegendes Verständnis für die wichtigsten Konzepte im Compilerbau. Wir schauen uns dazu relevante aktuelle Tools und Frameworks an und setzen diese bei der Erstellung eines kleinen Compiler-Frontends für `Min-Python` ein.

Überblick Modulinhalte

1. Lexikalische Analyse: Scanner/Lexer
 - Reguläre Sprachen
 - Generierung mit ANTLR
2. Syntaxanalyse: Parser
 - Kontextfreie Grammatiken (CFG)
 - LL-Parser (Top-Down-Parser)
 - Generierung mit ANTLR
3. Semantische Analyse: Attributierte Grammatiken und Symboltabellen
 - Namen und Scopes
 - Typen, Klassen, Polymorphie
4. Zwischencode: Intermediate Representation (IR), Builder
5. Interpreter: AST-Traversierung

Zur Dokumentation und Praktikum sind Links zu den entsprechenden Lernressourcen verlinkt.

Lernmodul Sitzung

ANTLR GENERIEREN

lexer: Erzeugen eines Token-Streams aus einem Zeichenstrom

Der Lernmodul beschreibt die Generierung eines Lexers mit ANTLR. Es wird erklärt, wie ein einfacher Lexer für die Erkennung von Wörtern und Zeichenketten funktioniert. Die Erstellung eines Lexers mit ANTLR wird detailliert erläutert, einschließlich der Verwendung von Regeln und Aktionen.

Lexer: Erzeugen eines Token-Streams aus einem Zeichenstrom

Aus dem Eingabe-`qual`-Text:

```
if (x >= 0) {  
    if (y <= 0) {  
        z = 0;  
    } else {  
        z = 1;  
    }  
}
```

LICENSE



Unless otherwise noted, this work is licensed under CC BY-SA 4.0.

See github.com/cagix/dlk23 for sources, slides and handout.