

Praktikum zu Hardwarenahe Programmierung

Prof. Dr. Stefan Conrad

Mitarbeiter: Thomas Germer,

Tutoren: Florian Dittrich, Larissa Knüpfer, Lars Leyendecker, Julian Ullrich, Nirojah Vettivel

Institut für Informatik
Heinrich-Heine-Universität Düsseldorf

Sommersemester 2021



Aufbau der Veranstaltung

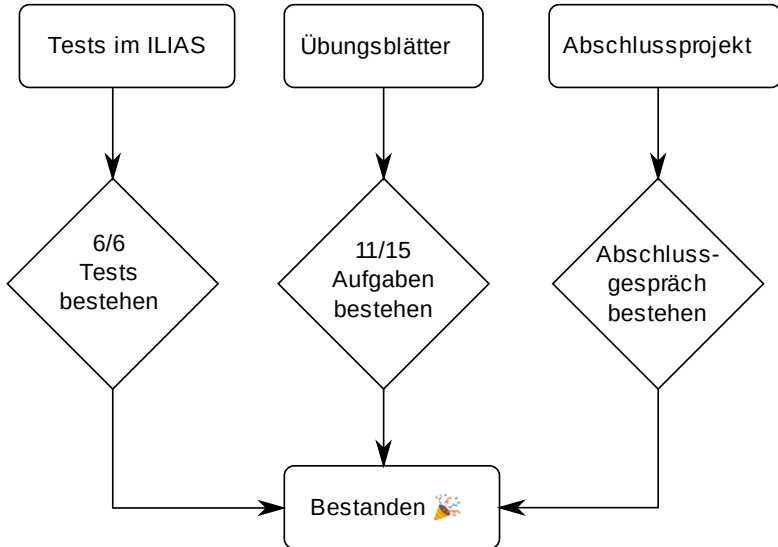
Die Veranstaltung besteht aus:

- 1 Vorlesungsblock zur Einführung
 - Videos dazu sind im ILIAS verlinkt.
- 2 Praktischer Programmierkurs
 - Ebenfalls im ILIAS.

Ablaufplan


- ➊ Anmeldung zu “Hardwarenahe Programmierung” im LSF
- ➋ Vortest im ILIAS bestehen (∞ Versuche)
- ➌ Anmeldung zu einer Übungsgruppe im ILIAS
- ➍ Übungen 1 bis 6:
 - ➊ Inhalte im Lernmodul/Buch erarbeiten
 - ➋ Übungsblatt lösen
 - ➌ Test im ILIAS bestehen
- ➎ Abschlussprojekt lösen
- ➏ Termin für Abschlussgespräch mit Tutor vereinbaren
- ➐ Abschlussgespräch bestehen

Kriterien zum Bestehen




- Vorlesungsaufzeichnungen, Forum, Vortest, Lernmodul, Tests, Gruppen und Termine sind im ILIAS zu finden.

INHALT




Vorlesungsaufzeichnungen




Forum - Hardwarenahe Programmierung

Beiträge (Ungelesen): 0 (0) Forum anonymisiert: Ja




Vortest

Notwendiger Test vor Gruppenanmeldung.




Lernmodul

Typ: Lernmodul ILIAS



Tests (verpflichtend bis spätestens zum 30.09.)



Gruppe 01

- Übungen finden statt über BigBlueButton (Link in Übungsgruppe).
- Um Fragen schon vor der Übung zu stellen: RocketChat
 - Gibt Tutoren Zeit, um bessere Antworten für die Übung zu organisieren.
 - Keinen Code im RocketChat posten! Führt sonst zu Plagiaten.



 **Gruppe 111**

Inhalt Info Einstellungen Mitglieder Lernfortschritt Metadaten Export Rec

[Zeigen](#) [Verwalten](#) [Sortierung](#) [Seite gestalten](#)

Link zum Videokonferenzsystem:
<https://bbb.cs.hhu.de/bj/...>

Link zur Rocketchat-Gruppe:
<https://rocketchat.hhu.de/invite/...>

ÜBUNGEN

 **Abgabe**

Stellen Sie sicher, dass Ihre Abgaben genau so, wie Sie sie abgegeben haben, kompilierbar sind!

Nächste Abgabefrist:

- Übungsgruppen wahlweise in einem von 3 Zeitblöcken:
 - ① 02.08.2021 – 25.08.2021 *oder*
 - ② 16.08.2021 – 13.09.2021 *oder*
 - ③ 03.09.2021 – 30.09.2021
- Voraussichtlich 8 Übungsgruppen je Zeitblock.
- Mehrere Uhrzeiten zur Auswahl.
- Uhrzeiten und Anmeldung zu Übungsgruppen ab 1. Juni im ILIAS.

- Beispiel

Tag	Datum	Ausgabe	Abgabe	Veranstaltung
Fr	03.09.2021	Blatt 1 (8:00)		
Mo	06.09.2021		Blatt 1 (23:55)	Übungsgruppe (8:30)
Mi	08.09.2021	Blatt 2 (8:00)		
Fr	10.09.2021		Blatt 2 (23:55)	Übungsgruppe (8:30)
Mo	13.09.2021	Blatt 3 (8:00)		
Mi	15.09.2021	Blatt 4 (8:00)	Blatt 3 (23:55)	Übungsgruppe (8:30)
Fr	17.09.2021	Blatt 5 (8:00)	Blatt 4 (23:55)	Übungsgruppe (8:30)
Mo	20.09.2021	Blatt 6 (8:00)	Blatt 5 (23:55)	Übungsgruppe (8:30)
Mi	22.09.2021	Projekt (8:00)	Blatt 6 (23:55)	Übungsgruppe (8:30)
Fr	24.09.2021			
Mo	27.09.2021			Übungsgruppe (8:30)
Mi	29.09.2021		Projekt (8:00)	
Do	30.09.2021			Abschlussgespräch

- Übungsblätter jeweils verfügbar ab 8:00 Uhr am Mo, Mi und Fr.
- Übungsgruppen ebenfalls Mo, Mi und Fr. Uhrzeit je nach Gruppe.
- Abgabe bis 23:55 Uhr des übernächsten Tages.

▼ ● Tag 1 (Verpflichtend)
Verbleibende Bearbeitungsdauer: 2 Tage, 15 Stunden, 51 Minuten | Abgabetermin: 05. Aug 2020, 23:55

DATEIEN

uebung1.zip	Download
-------------	----------

TERMINPLAN

Startzeit	Heute, 08:00
Abgabetermin	05. Aug 2020, 23:55
Verbleibende Bearbeitungsdauer	2 Tage, 15 Stunden, 51 Minuten

IHRE EINREICHUNG

Abgegebene Dateien	Sie haben noch keine Datei abgegeben.
	<button>Datei abgeben</button>

- Jedes Übungsblatt hat zwei bis drei Pflichtaufgaben.
- Mindestens 11 von insgesamt 15 Pflichtaufgaben müssen fehlerfrei gelöst werden.
 - Besser: Alle Aufgaben lösen. Abschlussprojekt sonst schwer.
 - Weitere optionale (nicht Pflicht-) Aufgaben zur Vorbereitung.
- Bei Problemen: Fragen in der Übungsgruppe stellen!

Abgabe der Übungsblätter

- Abgaben müssen im ILIAS hochgeladen werden.
- Lösungen in ein **ZIP**-Archiv packen und dann im ILIAS hochladen.
 - Andere Formate (z.B. bz2, gz, rar, tar, 7z, etc.) werden nicht akzeptiert!
 - Dateiendung umbenennen zählt nicht!
- Text-Kodierung: Nur UTF-8!
 - Kein UTF-16, ISO-8859-1, etc.
- Abgaben werden automatisiert getestet.
 - Dateien dürfen **nicht** umbenannt werden!
 - Namen von Unterordnern ebenfalls beibehalten.
 - Namen von Oberordnern und ZIP-Archiv sind unwichtig.
 - Bei mehreren Abgaben mit gleichem Dateinamen wird eine zufällige Datei ausgewählt und bewertet.
 - Dateien müssen genau so wie hochgeladen kompilierbar sein!

Abgabe der Übungsblätter

- Den Button mit der Beschriftung “Mehrere Dateien als ZIP-Archiv hochladen” **nicht** verwenden!
 - Verändert Dateinamen (z.B. *test.sh* zu *testsh.sec*).
 - Verwirft Ordnerstruktur.













The screenshot shows a web interface for submitting assignments. At the top, there is a header with a document icon and the word "Übung". Below this is a navigation bar with a back arrow and the text "Zurück", and a button labeled "Einreichung". The main content area has two buttons: "Datei hochladen" and "Mehrere Dateien als ZIP-Archiv hochladen". The second button is crossed out with a large red 'X'. Below these buttons, there is a section titled "BEREITS ABGEGEBENE DATEIEN - TESTDATEI". This section contains a list of files: "testsh.sec" and "testts.sec". Each file has a checkbox to its left and a "Löschen" button to its right. Red wavy lines are drawn under the file names "testsh.sec" and "testts.sec". Green arrows point to the "Zurück" button, the "Datei hochladen" button, and the "Löschen" button under "testts.sec".


Fragen stellen

- Bei Programm-Fehlern:
 - Reduzieren Sie ihr Programm so weit wie möglich, bis ihr Problem nicht mehr auftritt. Oft wird die Ursache dadurch schon offensichtlich.
 - `gcc -fsanitize=address -fsanitize=undefined` und `valgrind` verwenden, um Fehler genau zu lokalisieren (Übung).
- Bei Logik-Problemen:
 - Eigene Tests schreiben, um Funktionsweise des Programms zu überprüfen. Programm so gliedern, dass Tests möglich sind.
 - Aufgabe auf Papier durchgehen, Zwischenergebnisse vergleichen.
- Bei Umsetzungsschwierigkeiten:
 - Inhalte im Lernmodul/Büchern/Internet recherchieren.
- *Danach:* Frage in Übungsgruppe stellen.

- Tests möglichst zeitnah zur Übung lösen und nicht erst am 30.09.
- Anzahl erlaubte Versuche: ∞

INHALT

	Vorlesungsaufzeichnungen	
	Forum - Hardwarenahe Programmierung Beiträge (Ungelesen): 0 (0) Forum anonymisiert: Ja	
	Vortest Notwendiger Test vor Gruppenanmeldung.	
	Lernmodul Typ: Lernmodul ILIAS	
	Tests (verpflichtend bis spätestens zum 30.09.)	
	Gruppe 01	



- Umfangreichere Übungsaufgabe
- Abschlussprojekt muss *vollständig* gelöst werden
- Abschlussprojekt muss im Abschlussgespräch erklärt werden
- Termin zum Abschlussgespräch muss mit Tutor vereinbart werden
 - Genauere Details dazu in der Übung

Tutorial: Ein C-Programm kompilieren

Programme installieren

Zeilenweise im Terminal ausführen:

```
sudo apt-get update  
sudo apt-get upgrade  
sudo apt-get install build-essential pkg-config  
sudo apt-get install make gpg gdb check valgrind unzip
```

Hello, World!

- 1 Datei `hello.c` mit folgendem Inhalt erzeugen:

```
#include <stdio.h>
int main(void){
    printf("Hello, World!\n");
    return 0;
}
```

- 2 Terminal öffnen, Datei `hello.c` zu Programm `hello` kompilieren:

```
gcc -Wall -std=c99 hello.c -o hello
```

- 3 Programm ausführen:

```
./hello
```

- 4 Auf Dauer weniger Arbeit: Mit Skript (später mit Make) kompilieren.

```
echo "gcc -Wall -std=c99 hello.c -o hello" > compile.sh
chmod +x compile.sh && ./compile.sh && ./hello
```

Beispiel

```

user@pc:~$ sudo apt-get update
Hit:1 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:2 http://de.archive.ubuntu.com/ubuntu focal InRelease
Hit:3 http://de.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:4 http://de.archive.ubuntu.com/ubuntu focal-backports InRelease
Reading package lists... Done
user@pc:~$ sudo apt-get upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
user@pc:~$ sudo apt-get install build-essential pkg-config
Reading package lists... Done
Building dependency tree
Reading state information... Done
pkg-config is already the newest version (0.29.1-0ubuntu4).
build-essential is already the newest version (12.8ubuntu1.1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
user@pc:~$ sudo apt-get install make gpg gdb check valgrind unzip
Reading package lists... Done
Building dependency tree
Reading state information... Done
make is already the newest version (4.2.1-1.2).
unzip is already the newest version (6.0-25ubuntu1).
check is already the newest version (0.10.0-3build2).
gdb is already the newest version (9.2-0ubuntu1~20.04).
gpg is already the newest version (2.2.19-3ubuntu2.1).
valgrind is already the newest version (1:3.15.0-1ubuntu9.1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
user@pc:~$ echo -e '#include <stdio.h>\nint main(void){\n    printf("Hello, World!\\n");\n    return 0;\n}\n' > hello.c
user@pc:~$ gcc -Wall -std=c99 hello.c -o hello
user@pc:~$ ./hello
Hello, World!

```