

# eezi

## Eine Einführung zum Informatikstudium am KIT

19.11.2024, 2. eezi Vorlesung

KIT – Fakultät für Informatik

# Inhalt

- Programmieren Best Practices
- Effiziente Verfahren, Ratschläge, Tipps für Studierende zum Umgang mit bestimmten Übungsblättern
- Ausblick für den Rest des Semesters
- Fragen und Antworten
- Lernpartnerschaftsbörse

**Es ist, was es ist**  
**Aber es wird, was du daraus machst**

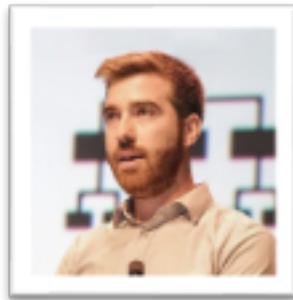
# PROGRAMMIEREN BEST PRACTICES

- Timur Sağlam

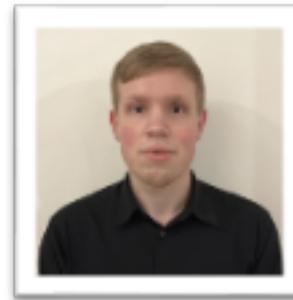
# Das Programmieren-Team



Dominik Fuchß



Timur Sağlam



Nils Niehues



Haoyu, Liu



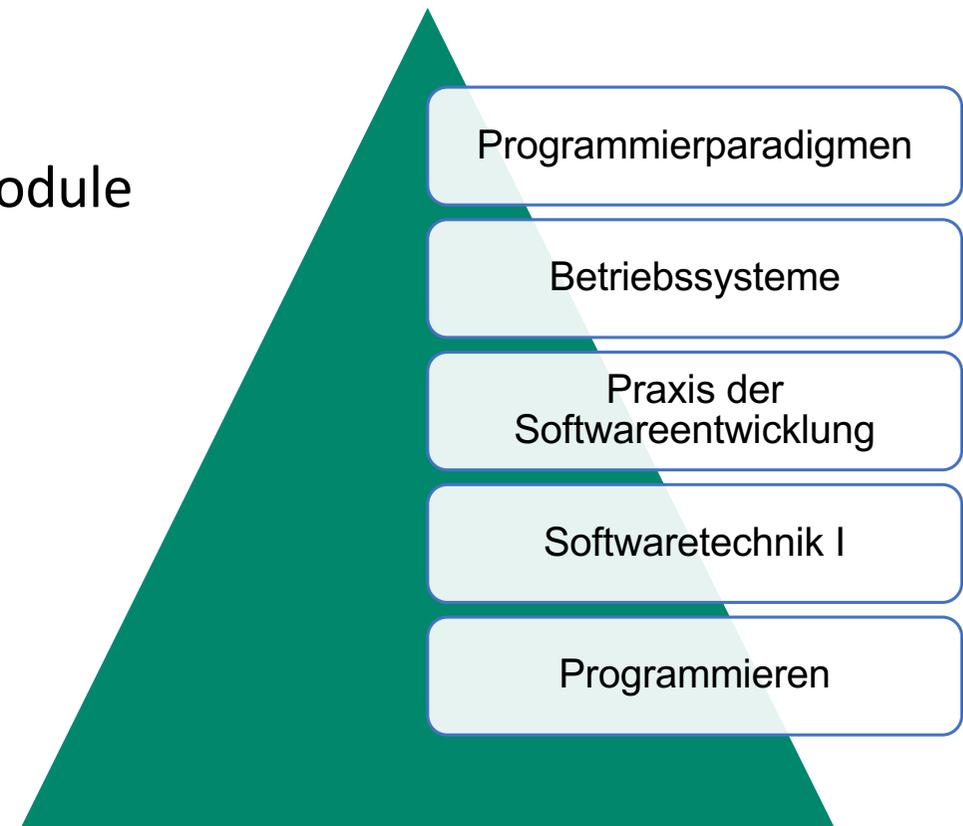
Robin Maisch

# Best Practices für „Programmieren“

- Heutige Ziele:
  - Woran kann es scheitern?
  - Wie Frust sparen?
  - Wie beste Chancen haben
  
- Nicht alle Kriterien werden bei den ersten Blättern anwendbar sein!
  - Es muss nicht alles *sofort* verstanden werden
  - Aber spätestens bei den Abschlussaufgaben

# Grundstudium: Praktische Informatik

- Programmieren...
  - ...bildet die Grundlage für weitere Module
  - ...ist Orientierungsprüfung
  - ...sollte vor SWT abgeschlossen sein
- Was kann man mit guten Programmierkenntnissen tun?
  - Studentische Hilfskraft
  - Werkstudent
  - Open-Source-Projekte



# Was Sie aus diesem Modul mitnehmen sollten

- Die Grundlagen der objektorientierten Programmierung in Java
- **Abbildung:** Problemmodellierung
  - Wie man alltägliche Probleme in Programmiersprache modelliert
- **Automatisierung:** Lösungsformulierung
  - Entwicklung von Verfahren (Algorithmen) zur Lösung einfacher Probleme
- **Abstraktion:** Problemformulierung
  - Wie man nicht nur eine Probleminstance löst, sondern eine allgemeine Lösungsmethode findet, die für viele Probleminstance funktioniert
- **Sauber programmieren!**
  - Wichtig für Wartbarkeit, Wiederverwendbarkeit und Zusammenarbeit



# Allgemeine Tipps zur Studienorganisation

- Termine und Fristen verfolgen
  - Bereits zu Beginn des Semesters veröffentlicht: [s.kit.edu/programmieren](https://s.kit.edu/programmieren)
  - In den eigenen Kalender mit Erinnerung übernehmen
  - Keine nachträglichen Anmeldungen für Prüfungsleistungen
  
- Studien- und Prüfungsordnung beachten
  - Arbeit muss selbstständig angefertigt werden
  - Der Übungsschein kann mehrfach wiederholt werden
  - Abschlussaufgaben können nach dem Übungsschein erworben werden
  - Die Abschlussaufgaben können nur einmal wiederholt werden
    - Muss bis zum Ende des dritten Semesters bestanden werden
    - Keine mündliche Nachprüfung

# Vorlesung und Übung nachbereiten

- Folien der Vorlesung während des Semesters aktiv nacharbeiten
  - Im Semester nacharbeiten und nicht kurz vor den Abschlussaufgaben
  - Aufzeichnungen der Vorlesung sind auf YouTube verfügbar
  - Vorlesungsfolien sind im ILIAS vorhanden: [s.kit.edu/ilias](https://s.kit.edu/ilias)
  - Übungsblätter orientieren sich an der Struktur der Vorlesung
- Beispiellösungen der Übungsblätter nachvollziehen
  - Geben Hinweise zur Bearbeitung der Abschlussaufgaben
  - Nur Beispiele dafür, wie die Aufgabe gelöst werden könnte
  - Verstehen der verwendeten Konzepte in den Beispiellösungen
    - Entwurfsentscheidungen versuchen zu verstehen
    - Das Vorgehen nachvollziehen und verstehen

# Unterstützung während des Semesters

## Tutorien

- Möglichkeit zum Nachfragen
- Übungsblatt vor- und nacharbeiten
- Vorbereitung auf Präsenzübung
- [s.kit.edu/programmieren](https://s.kit.edu/programmieren)

## Info<sup>2</sup>

- Info<sup>2</sup> und Lerngruppenbörse
- Mentoring und Kennenlernen Anderer
- Fachliche und persönliche Unterstützung
- Studienplanung Hilfsmittel

## MINT

- Wiederholung des Vorlesungsinhalt
- Begrenzte Teilnehmerzahl
- Aufzeichnungen im ILIAS
- [s.kit.edu/mint](https://s.kit.edu/mint)

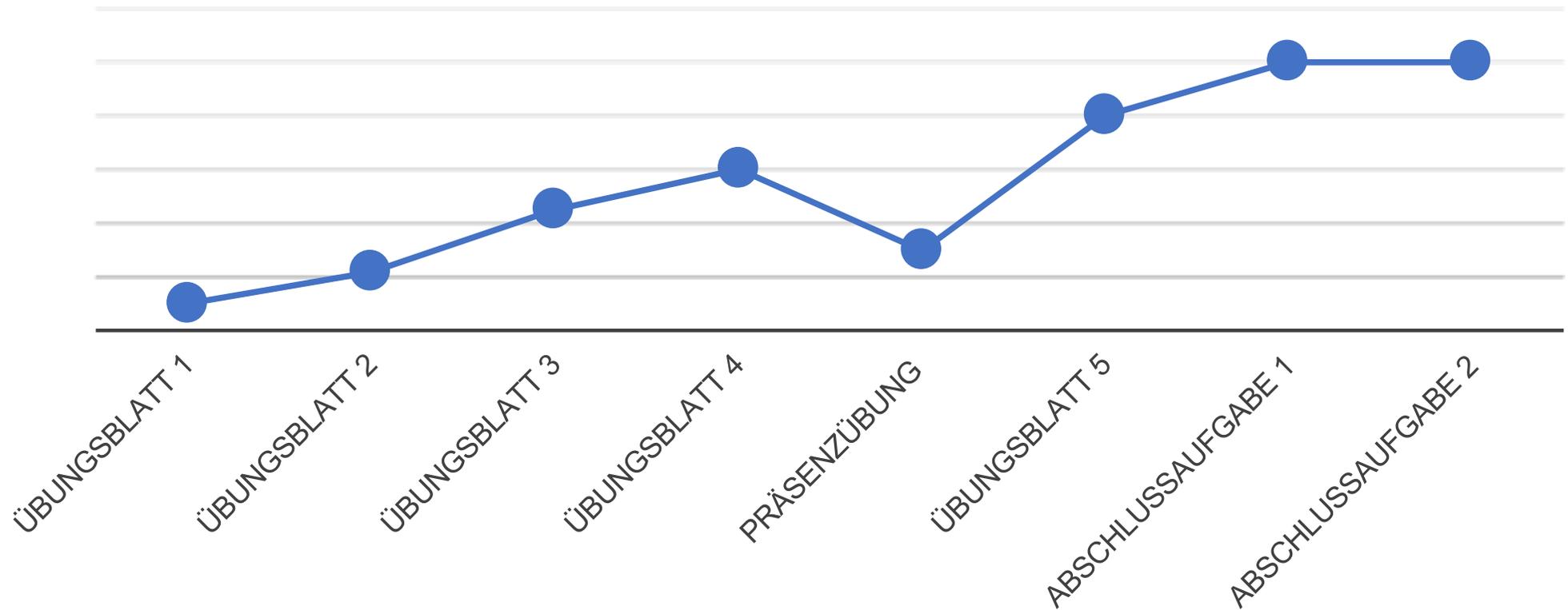
## Forum-Wi / (Aktive) Fachschaft

- Vielfältigen Angebot auf dem Campus
- Klausuren und Prüfungsprotokollen
- Fachliche und persönliche Unterstützung
- [s.kit.edu/fsmi](https://s.kit.edu/fsmi)      ■ [s.kit.edu/forum](https://s.kit.edu/forum)

# Programmieren Präsenzübung

- Prüft Wissen, welches bei der Bearbeitung der Übungsblätter erworben worden sein sollte
  - Kleine Quelltextausschnitte schreiben oder ergänzen
  - Verhalten von kleinen Quelltextausschnitten analysieren
  - Verbeißen Sie sich nicht zu lange in eine einzige Aufgabe
- Sie sind bereits gut vorbereitet, wenn Sie die Übungsaufgaben gewissenhaft und erfolgreich gelöst haben!
  - Kein besonderes Hintergrundwissen nötig
  - Keine Vorgaben zu Checkstyle-Regeln
- Zur Vorbereitung auf das „Programmieren auf Papier“
  - Frischen Sie die Java-Syntax auf
  - Nicht die Folien auswendig lernen

# Subjektive Schwierigkeit



# Übungsblätter und Abschlussaufgaben

## Übungsschein nicht erworben?

- Vorlesung beginnt bei null
- Übungsblätter orientieren sich an der Vorlesung
- Sehr steile Lernkurve
  - Sehr anspruchsvoll, wenn keine Vorkenntnisse vorhanden sind
  - Fortgeschrittene Studenten sollten für die Methodik am Ball bleiben

## Übungsschein bereits erworben?

- Übungsblätter als erneute Vorbereitung nutzen
- Das Abgabesystem anschauen
- Programme selbstständig implementieren
  - Aufgaben aus dem Internet
  - Algorithmen aus den Mathematikvorlesungen

# Organisatorische Tipps für die Übungsblätter

- Übungsblätter als Vorbereitung nutzen
  - Durch das eigenständige Bearbeiten lernt man am meisten
  - Praktisch fast kein zusätzlicher Lernaufwand für die Präsenzübung
  - Das letzte Übungsblatt als Vorbereitung auf die Abschlussaufgaben nutzen
- Früh genug anfangen, die Übungsblätter zu bearbeiten
  - Unklarheiten können so frühzeitig beseitigt werden
  - Die Übungsblätter werden schnell deutlich anspruchsvoller
  - Auf den ersten Blättern so viele Punkte wie möglich sammeln
- Früh genug anfangen, die Übungsblätter abzugeben
  - Frühzeitiges erproben von Git und Artemis
  - Lösungen können beliebig häufig hochgeladen werden
  - Tutoren sieht nur die letzte gültige Abgabe

# Praktische Tipps für die Aufgaben

- Quelltextklone vermeiden
  - Sich nicht Wiederholen und alles nur einmal im Quelltext implementieren
  - Aus doppelten Quelltextstellen versuchen, Methoden zu extrahieren
- Lange Methoden vermeiden
  - Methoden zerteilen und auf mehrere kleine (private) Hilfsmethoden verteilen
  - Dokumentation der Hilfsmethoden durch Javadoc-Kommentare
- Over-Engineering vermeiden
  - Quelltext so einfach wie möglich halten
  - Keine nicht benötigte Funktionalität
- Trennung der Anliegen
  - Datenstrukturen nicht nach außen geben
  - Trennung von Anwendungslogik und Benutzungsschnittstelle
- Geeignete Datentypen verwenden
  - Enums bei abgeschlossenen Mengen
  - Primitive Datentypen verwenden
  - Modellierung komplexer Typen mit Klassen
- Herangehensweise
  - Aufgabestellung genau lesen
  - Exakt die erwartete Ausgabe liefern
  - Bei Unklarheiten sofort nachfragen

## Methodik Abzüge

- Neben Funktionalität wird auch Methodik geprüft
- Großer Teil der Bewertung von Übungsblättern und Abschlussaufgaben
- Alle Bewertungskriterien sind im Wiki: [sdq.kastel.kit.edu/programmieren](https://sdq.kastel.kit.edu/programmieren)

*(zuvor im Ilias Kurs)*

# Ilias-Wiki

## Materialsammlung

**Vorlesungsmaterial**  
Foliensätze und Aufzeichnungen der Vorlesung.

**Tutoriumsmaterial**  
Inhalte aus den Tutorien.

**Übungsblätter**  
Aufgabenstellungen, Abgabe und weitere Materialien zu den Übungsblättern.

**Programmieren-Wiki**  
Erste Schritte, Methodik und Bewertungsrichtlinien.

**Vorlesungsaufzeichnungen WiSe 24/25**  
Kursnummer: WS242424004 ; Ilias ID: 2476137 ; Semester: WS 24/25 ; CampusID

# Ilias-Wiki: Übersicht

## Hauptseite

[Hauptseite](#) [Diskussion](#)

[Lesen](#) [Formular anzeigen](#) [Quelltext anzeigen](#) [Versionsgeschichte](#)

## Beschreibung

Dieses Wiki wurde im Sommersemester 2024 erstellt und erweitert, das vorausgegangene Ilias-Wiki, das ab 2013/14 von Lehrenden für diese Vorlesung erstellt und instand gehalten wurde.

Aufgrund dessen, dass dieses Wiki sehr neu ist, kann es sein, dass sich Fehler eingeschlichen haben, die bislang nicht gefunden und korrigiert wurden. Diese dürfen gerne über das [Artemis-Forum](#) weitergeleitet werden, dass diese dann zeitnah korrigiert werden. Auch Ergänzungswünsche sind gerne gesehen, seien es zusätzliche Informationen oder gar neue Artikel. Das Wiki lebt von dem Feedback der Studierenden und wird dadurch nur besser.

Zur Installation von Java s. [Java](#).

Im Folgenden werden die aus unserer Sicht wichtigsten, allerdings nicht alle, Artikel aufgelistet (diese sind dann über die Suchleiste oben zu erreichen):

<b>Organisation</b>	<a href="#">[Einklappen]</a>	<b>Grundlagen</b>	<a href="#">[Einklappen]</a>	<b>Bewertungsrichtlinien</b>	<a href="#">[Einklappen]</a>
<ul style="list-style-type: none"><li>– <a href="#">Artemis</a></li><li>– <a href="#">Beratungsstellen</a></li><li>– <a href="#">FAQ</a></li><li>– <a href="#">Nutzung Testfälle</a></li><li>– <a href="#">Präsenzübung</a></li></ul>		<ul style="list-style-type: none"><li>– <a href="#">Abstrakt</a></li><li>– <a href="#">Algorithmik</a></li><li>– <a href="#">Checkstyle</a></li><li>– <a href="#">Command Pattern</a></li><li>– <a href="#">Datentypen</a></li><li>– <a href="#">Debugging</a></li><li>– <a href="#">Dokumentation</a></li><li>– <a href="#">Enum</a></li><li>– <a href="#">Exceptions Kontrollfluss</a></li></ul>		<b>Blatt 1</b> <ul style="list-style-type: none"><li>– <a href="#">Dokumentation</a></li><li>– <a href="#">Einheitliche Sprache</a></li><li>– <a href="#">Komplexität</a></li><li>– <a href="#">Leerer Block/Leerer Konstruktor</a></li><li>– <a href="#">Nur Main</a></li><li>– <a href="#">Scanner</a></li><li>– <a href="#">Schlechter Bezeichner</a></li><li>– <a href="#">Schwieriger Code</a></li></ul>	

# Ilias-Wiki: Bewertungsrichtlinien

## Bewertungsrichtlinien [Einklappen]

### Blatt 1

- Dokumentation
- Einheitliche Sprache
- Komplexität
- Leerer Block/Leerer Konstruktor
- Nur Main
- Scanner
- Schlechter Bezeichner
- Schwieriger Code
- Unbenutztes Element
- Verwendung von instanceof

### Blatt 2

- Assertions
- Bedeutungslose Konstanten
- Enum
- Fehlermeldungen
- Final
- Getter / Setter für Listen
- Hartcodieren

- Hilfsklasse
  - IO/UI
  - JavaDoc
  - JavaDoc Trivial
  - Konstanten-Klasse
  - Magic Number
  - Object statt konkreter Klasse
  - Pakete
  - Polymorphie
  - Raw Types
  - Reimplementierung
  - Runtime Exceptions
  - Sichtbarkeit
  - Stringreferenzen
  - Ungeeigneter Schleifentyp
  - Zeilenumbrüche
- ### Blatt 3
- Duplikate
  - Exceptions Kontrollfluss
  - Große Try-Catch Blöcke
  - Interface statt konkreter Klasse
  - Statische Methoden und Attribute
  - Verweigte Vererbung



Alle gelten für die  
Abschlussaufgaben!

# Ilias-Wiki: Einheitliche Sprache

Seite [Diskussion](#) [Lesen](#) [Mit Formular bearbeiten](#) [Bearbeiten](#) [Quelltext bearbeiten](#) [Versionsgeschichte](#) [★](#) [Weitere](#) [v](#)

 Diese Seite ist eine Bewertungsrichtlinie, die ab Blatt 1 annotiert und ab Blatt 1 abgezogen wird. 

**Beschreibung** [\[ Bearbeiten | Quelltext bearbeiten \]](#)

Es kann sehr anstrengend für eine Person werden, die den Quellcode verstehen möchte, Kommentare in unterschiedlichen Sprachen zu lesen. Auch gibt es das Problem der Sprachbarriere. Kommentare in einer unbekanntenen Sprache sind demnach unverständlich und nicht hilfreich. Sind Kommentare in einheitlicher, teils vorgegebener Sprache, kann sichergestellt werden, dass alle in dem Projekt involvierten Personen in der Lage sind, die Kommentare zu lesen und verstehen. Für Kommentare und Ein-/Ausgabe gibt es verschiedene Regelungen:

**JavaDoc und Kommentare**

JavaDoc und Entwicklerkommentare sollen in diesem Übungsbetrieb auf Deutsch oder Englisch verfasst werden. Hierbei soll es über das gesamte Projekt einheitlich sein. Auch zwischen verschiedenen Kommentartypen. Welche Sprache das ist, darf selbst ausgesucht werden.

**Ein-/Ausgabe**

Auch für die Ein-/Ausgabe soll auch eine einheitliche Sprache gewählt werden. Für Ausgaben ist zu beachten, dass diese, wie andere String-Literale auch, als Programmcode zählen und demnach auf Englisch verfasst werden sollten. Es allerdings Aufgaben geben, in denen eine deutsche Ausgabe gefordert wird. In diesem Fall sind die Ausgaben auf Deutsch zu verfassen.

**Randnotiz**

Die Ausgabe und JavaDoc/Kommentare dürfen in unterschiedlicher Sprache verfasst werden. Sprich: Deutsche Kommentare mit englischer Ausgabe sind okay, sofern die Aufgabenstellung keine deutsche Ausgabe verlangt. Auch ist zu beachten, dass sich diese Regel wirklich nur auf Kommentare und JavaDoc bezieht. Quellcode und demnach auch Bezeichner sollen nach wie vor auf Englisch verfasst werden (siehe [Schlechter Bezeichner](#)).

Das wäre ein **leichtes Vergehen**.

# Ilias-Wiki: Einheitliche Sprache

## 1 Negativbeispiel

```
1  /**
2  * Prints the status of the player to the console.
3  * @param player The id of the player
4  */
5  public void printPlayerStatus(int playerId) {
6      // Falls der Spieler unbekannt ist wird null zurückgegeben
7      Player player = findPlayerById(playerId);
8
9      if (player == null) {
10         System.err.println("Error, unbekannter Spieler");
11     } else {
12         System.out.println("Player " + playerId + ": " + player.getHP() + "HP");
13     }
14 }
```

# Ilias-Wiki: Einheitliche Sprache

## 2 Positivbeispiel

```
1  /**
2  * Prints the status of the player to the console.
3  * @param player The id of the player
4  */
5  public void printPlayerStatus(int playerId) {
6      // Returns null if the player is unknown
7      Player player = findPlayerById(playerId);
8
9      if (player == null) {
10         System.err.println("Error, unknown player");
11     } else {
12         System.out.println("Player " + playerId + ": " + player.getHP() + "HP");
13     }
14 }
```

## Schlechte Bezeichner

- Bezeichner sollten *sprechend* sein
- Keine Abkürzungen, konform zu Konventionen
- Ausnahmen: i, j, x, y, it

```
1 public static int s(final int a, final int b) {  
2     return a - b;  
3 }
```



```
1 public static int subtract(final int minuend, final int subtrahend) {  
2     return minuend - subtrahend;  
3 }
```



## Final

- Attribute sollten immer, wenn möglich, final sein
- Verhindert unabsichtliche Änderungen der Attribute (nicht ihrer Zustände)

```
class Event {  
    private static int BASE_FEE = 1000;  
    private List<String> participants;  
    public Event() {  
        participants = new ArrayList<>();  
    }  
    public void addParticipant(String name) {  
        participants.add(name);  
    }  
    public int calculateProfit(int ticketPrice, int venueCost) {  
        int fixedCost = BASE_FEE + venueCost;  
        return participants.size() * ticketPrice - fixedCost;  
    }  
}
```



## Final

- Attribute sollten immer, wenn möglich, final sein
- Verhindert unabsichtliche Änderungen der Attribute (nicht ihrer Zustände)

```
class Event {  
    private final static int BASE_FEE = 1000; // constants should be final  
    private final List<String> participants; // fields should be final if possible  
    public Event() {  
        participants = new ArrayList<>();  
    }  
    public void addParticipant(String name) { // not required for parameters or local variables  
        participants.add(name); // (list is final, but not the content)  
    }  
    public int calculateProfit(int ticketPrice, int venueCost) {  
        int fixedCost = BASE_FEE + venueCost;  
        return participants.size() * ticketPrice - fixedCost;  
    }  
}
```



## Ungeeigneter Schleifentyp

- Boolescher-Ausdruck => While-Schleife, Do-While-Schleife
- Einfache Iteration => For-Each-Schleife
- Index benötigt => For-Schleife

```
1 public void printList(List<String> list){  
2     for(int i = 0; i<list.size(); i++){  
3         System.out.println(list.get(i));  
4     }  
5 }
```



```
1 public void printList(List<String> list){  
2     for(String item : list){  
3         System.out.println(item);  
4     }  
5 }
```



# Unnötige Komplexität

- Redundanz/Komplexität erhöht visuelles Rauschen  
=> einfachste Lösung wählen

```
1  boolean isValid() {  
2      if (this.sold == false && !(this.price <= 0)) {  
3          return true;  
4      } else {  
5          return false;  
6      }  
7  }
```



```
1  boolean isValid() {  
2      return !this.sold && this.price > 0;  
3  }
```



## Magic Numbers/Strings

- Alleinstehende Zahlen im Quelltext sind oft schwer zu verstehen
- Variablen erlauben Benennung
- Konstanten erlauben Wiederverwendung

```
1 final String password = "Test";  
2 if (password.length() < 8) {  
3     System.err.println("Password too short, please choose a longer password!");  
4 }
```



```
1 final String password = "Test";  
2 final int minimalPasswordLength = 8;  
3 if (password.length() < minimalPasswordLength) {  
4     System.err.println(PASSWORD_ERROR_MESSAGE);  
5 }
```



# Sichtbarkeit und Datenkapselung

- Die Sichtbarkeit sollte so restriktiv wie möglich gesetzt werden
- Interner Zustand soll nur, falls wirklich nötig, herausgegeben werden
- Die Möglichkeit zu ungeplanten Zugriff führt zu Bugs

```
public final class Book {  
    public String title;  
    long isbn;  
  
    protected Book(String title, long isbn) { // protected sinnlos, da Klasse final  
        if (!isIsbnValid(isbn)) {  
            throw new InvalidIsbnException("The isbn is not valid");  
        }  
        this.title = title;  
        this.isbn = isbn;  
    }  
  
    public boolean isIsbnValid(long isbn) { // interne Hilfsmethode  
        // ...  
    }  
}
```



# Sichtbarkeit und Datenkapselung

- Die Sichtbarkeit sollte so restriktiv wie möglich gesetzt werden
- Interner Zustand soll nur, falls wirklich nötig, herausgegeben werden
- Die Möglichkeit zu ungeplanten Zugriff führt zu Bugs

```
public final class Book {  
    private String title;  
    private long isbn;  
    public Book(String title, long isbn) {  
        if (!isIsbnValid(isbn)) {  
            throw new InvalidIsbnException("The isbn is not valid");  
        }  
        this.title = title;  
        this.isbn = isbn;  
    }  
    public String getTitle() { return this.title; }  
    public long getIsbn() { return this.isbn; }  
    private boolean isIsbnValid(long isbn) {  
        // ...  
    }  
}
```



# Testen

- Testen hilft, Fehler zu finden!
  - Vergleich von tatsächlichem und erwünschtem Verhalten
  - Modularer Quelltext hilft beim Testen
- Ausführliches Testen vordem Hochladen
  - Nicht auf den Public-Test verlassen
  - Den Public-Test lokal ausführen
- Selber Tests schreiben
  - Für die Kommandozeileninteraktion
  - Über eigene Main-Methode
  - JUnit-Tests schreiben

## Teststrategien

- Datenbasiert
  - Beispiele auf Aufgabenblättern
- Kontrollflussbasiert
  - Suche in Datenstrukturen
  - Alle Entscheidungen einmal treffen
- Grenzwertbasiert
  - Wertebereich bei Berechnungen
  - Off-by-one-Error

# Rückfragen

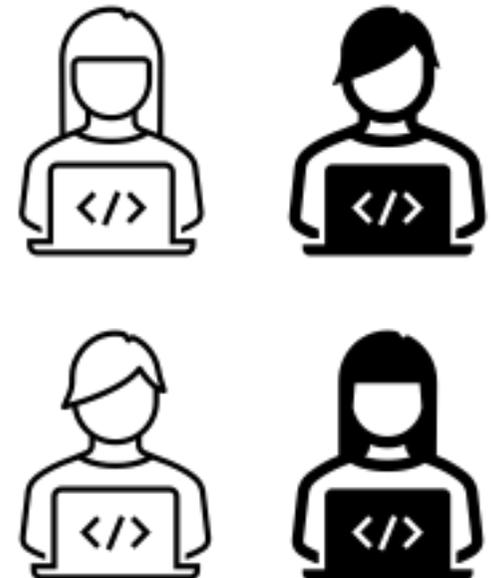
- Fragen und Unklarheiten können immer auftreten
  - Wenn ihr eine Frage habt, haben andere vielleicht die gleiche Frage
  - Wir bieten viele Möglichkeiten, Fragen zu stellen
- Bei Rückfragen werden Aufgaben gegebenenfalls aktualisiert
  - Lediglich minimale Aktualisierungen oder weitere Klarstellungen
  - Schaut immer mal wieder, ob die aktuelle Aufgabe aktualisiert wurde

## Reihenfolge bei Fragen

1. Websuche
2. FAQ / Programmieren-Wiki prüfen
  - [s.kit.edu/faq](https://s.kit.edu/faq)
  - [sdq.kastel.kit.edu/programmieren](https://sdq.kastel.kit.edu/programmieren)
3. Diskussionsforum prüfen
  - Suchfunktion verwenden
4. Im Diskussionsforum nachfragen
  - Aussagekräftigen Titel verwenden
5. Eure Tutoren fragen
6. Übungsleitung fragen
  - [programmieren-vorlesung@cs.kit.edu](mailto:programmieren-vorlesung@cs.kit.edu)

# Zusammenfassung

- Nehmen Sie sich **ausreichend Zeit** für die Übungsblätter
- Fangen Sie **rechtzeitig** mit der Bearbeitung an
- Beachten Sie die **Bewertungskriterien** im Ilias-Wiki
- Verwenden Sie früh **Checkstyle**
- Besuchen Sie die **Tutorien** und nehmen Sie aktiv Teil
- Üben Sie für die Präsenzübung **auf Papier**



# EFFIZIENTE VERFAHREN, RATSCHLÄGE, TIPPS FÜR STUDIERENDE ZUM UMGANG MIT BESTIMMTEN ÜBUNGSBLÄTTERN

# Übungsscheine Übersicht

Modul	Übungsschein notwendig für Prüfung	Semester
Grundbegriffe der Informatik	Nein aber um Modul zu bestehen	WS
Programmieren	Ja	WS + SS
Lineare Algebra I (für Info)	Ja	WS + SS: LA II Übungsschein
Lineare Algebra I (Mathe)	Ja	WS
Höhere Mathematik I	Ja	WS + SS: HM II Übungsschein
Analysis I	Ja	WS
Mathe I	Nein aber um Modul zu bestehen	WS
Wirtschaftsinformatik I	Nein nur Klausurbonus	WS

# GRUNDBEGRIFFE DER INFORMATIK

# Grundbegriffe der Informatik: Übungsschein

- Zum bestehen notwendig:
  - Mindestens **50%** der Punkte
  - Mindestens **20%** der Punkte auf jedes Aufgabenblatt (bis auf 2)
- Nur im WS machbar
  - Erst im 3. Semester nachholbar
- Übungsschein ist notwendig, um das Modul zu bestehen
- Prüfung lässt sich auch ohne Übungsschein schreiben

# Grundbegriffe der Informatik: Tipps

- Tutorium bereitet auf neues Übungsblatt vor
- Definitionen bereithalten (z.B. aus Skript)
- Nach ähnlichen Aufgaben und Lösungen suchen
  - Im Skript
  - In Übungsblättern der letzten Jahre
  - In Altklausuren
  - Illias, (Fachschaft)

# Grundbegriffe der Informatik: Hilfreiche Seiten

- Youtube 
- [NLogSpace](#)
- [Morpheus Tutorials](#)
  
- Websites
- [Fuks Crashkurse](#)

# MATHEMATIK

# Lineare Algebra I: Übungsschein

- Zum Bestehen notwendig:
  - Mindestens **40%** der Punkte jeweils in Übungsblättern 1-7 und 8-14
- Wird nur im WS angeboten
  - Alternative im Sommersemester Lineare Algebra II Übungsschein
  - MathematikerInnen brauchen beide Scheine
- Für das Orientierungsmodul notwendig
- Für die Prüfung notwendig

# Höhere Mathematik I: Übungsschein

- Zum Bestehen notwendig:
  - Mindestens **50%** der Punkte jeweils in Übungsblättern 1-7 und 8-13
- Wird nur im WS angeboten
  - Alternative: Höhere Mathematik II Übungsschein im Sommersemester
- Für die Prüfung notwendig

# Analysis I: Übungsschein

- Zum Bestehen notwendig:
  - Mindestens **40%** der Punkte jeweils in Übungsblättern 1-7 und 8-14
- Wird nur im WS angeboten
  - Alternative: Teilnahme an Scheinklausur, diese findet jedoch parallel zur Hauptklausur Analysis I statt
- Für die Prüfung notwendig

# Äquivalenz von Übungsscheinen

- Die Übungsscheine von **LA1 für Informatiker** und **LA1** sind gleichwertig und ein Wechsel ist möglich.
- Die Übungsscheine von **Analysis 1** und **Höhere Mathematik 1** sind gleichwertig und ein Wechsel ist möglich.
- Genauere Informationen findet man im Modulhandbuch oder im ISS FAQ.

# Mathe I: Übungsschein

- Zum Bestehen notwendig:
  - Mindestens **50%** der Punkte jeweils in Übungsblättern 1-7 und 8-15
- Wird nur im WS angeboten
  - am besten jetzt versuchen, da O-Prüfung
- Zum Bestehen des Moduls notwendig
- Teilnahme an Prüfung auch ohne Übungsschein möglich

# Mathematik: Tipps



- Sich **nicht** daran **aufhalten**, es sich **vorstellen** zu wollen
  - Mathematik an Uni ist sehr abstrakt
  - Versuchen es einfach mal nur stur anzuwenden



- **Ähnliche Aufgaben** mit Lösungen googlen, erfragen, ...
  - Lösungen aber *nicht* abschreiben, sondern nachvollziehen und auf Aufgabenstellung übertragen

# Mathematik: Hinweise



- Prüfungsaufgaben ähnlich zu Übungsblättern
  - Rechenaufgaben genau anschauen
  - Sehr viel üben!

# Mathematik

- Youtube 
- [3Blue1Brown](#) (LA, HM, Mathe I, Analysis I)
- [Mathepeter](#) (LA, HM, Mathe I, Analysis I)
- [blackpenredpen](#) (HM, Mathe I, Analysis I)
- [Mathe by Daniel Jung](#) (LA, HM, Mathe I, Analysis I)

# WIRTSCHAFTSINFORMATIK I

# Wirtschaftsinformatik I: Übungsschein

- Zum Bestehen notwendig:
  - Mindestens **75%** der Punkte
  - Alle 5 Übungsblätter müssen bearbeitet werden
- Wird nur im WS angeboten
- Erfolgreiches bestehen des Übungsscheins gibt einen Notenbonus

# Wirtschaftsinformatik I: Tipps

- Notenbonus mitnehmen
- Auswendig lernen!!

# ALLGEMEINE TIPPS

# Lerntechniken

Lerntechnik	Modul	Beispiel
Karteikarten (Anki)	Alle	Definitionen lernen
<a href="#">Pomodoro</a>	Alle	Hilft gegen Prokrastination
Deep Work	Mathe, Programmieren	Tief in Themen eintauchen
Feynman-Technik	Alle	Wissenslücken erkennen, Themen zusammenfassen
Spaced Repetition	Alle	Wissen verfestigen
Visualisieren (Mindmapping, Übersichten)	Winfo, GBI, Mathe	Überblick bekommen und Zusammenhänge verstehen
Distraction Delay	Alle	Fokussiert bleiben
Üben, üben, üben	Alle	Verständnis entwickeln

 <b>Organisieren und Planen</b>	 <b>Recherche und Textbearbeitung</b>	 <b>Struktur und Gedächtnis</b>	 <b>Motivation und Selbstregulation</b>	 <b>Entspannung und Bewegung</b>
To Do-Liste, Tagesplan, Masterplan erstellen	Internetrecherche (Suchmaschinen)	Mind-Mapping	Selbstmotivierung	Bewegung und Lernen
Meilensteinplanung	Kooperatives Lesen	Strukturlegetechnik	Pomodoro Technik	Bewegungspausen gestalten
Planung mit der ALPEN-Methode	Markierungstechniken	Lernposter erstellen	Prokrastination Selbsttest	Jonglage
Planung mit dem Kanban-Kalender	Lesetechniken	Mnemotechnik: Schlüsselwort-Methode	Aufschiebetagebuch führen	Stressbewältigung
Prioritäten setzen mit der Eisenhower-Matrix	Lesemethode PQ4R (für komplexe Texte)	Mnemotechnik: Zahlentafel/Gedächtnistafel	Umgang mit Lampenfieber	Entspannungsübungen
SMARTe Lernziele setzen	Speed Reading/ Schnelles Lesen (Selbsttest)	Mnemotechnik: Akronyme	Survival Guide Prüfungsangst	Achtsamkeit
Pausenplanung		Mnemotechnik: Loci-Methode	Survival Guide Gruppenarbeit	Powernapping
Lernplan erstellen		Mnemotechnik: Karteikarten-Methode	Umgang mit Erfolg und Misserfolg	
Lerntagebuch schreiben		Vorlesungsmitschrieb, Vor- und Nachbereitung	Emotionsregulation: Das ABC-Schema	
Der häusliche Arbeitsplatz		Vorlesungsmitschrieb: Cornell Methode	Ressourcen aktivieren	
Strukturiert im Homeoffice		Deep Work	Selbstregulierung mit der Affektbilanz	
Survival Guide Prüfungsphase		Hubschrauber-Landkarten-Methode	Selbststeuerung mit dem Wenn-Dann-Plan	



# Materialsammlungen

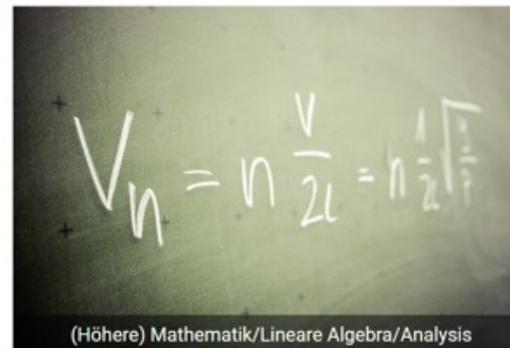
- Zusammenfassungen von Studierenden aus höheren Semestern
  - Studydrive
  - KIT Mathe Info Discord
  
- Vorlesung verpasst?
  - Lineare Algebra I Videos im Ilias von WS 20/21
  - Grundbegriffe der Informatik Videos im Ilias
  - Programmieren Videos im Ilias

# MINT-Kolleg

■ [MINT-Kolleg](#): Vorkurse, Semesterkurse, Helpdesk, Aufbaukurse

■ Bietet Semesterkurse an zu

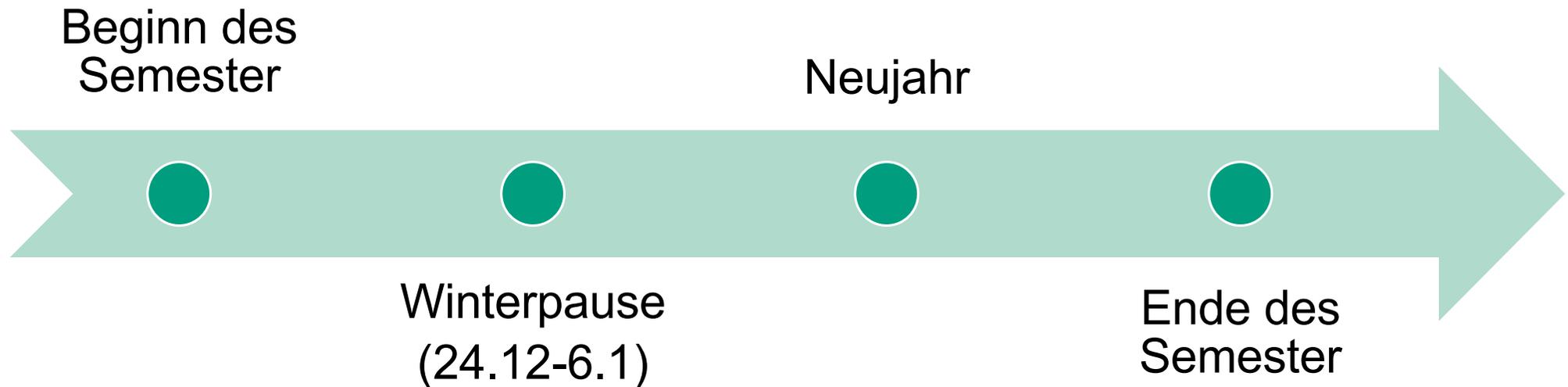
- Höhere Mathematik I
- Lineare Algebra I
- Programmieren



■ Im Ilias Kurs können Ressourcen genutzt werden, auch wenn man keinen Kursplatz hat

# REFLEXION

# Reflexion und Motivation



## ■ Gut Reflektiert ist halb Gewonnen

- „Passt ein Studium zu mir?“
- „Passt der Studiengang zu mir?“
- „Was habe ich bisher erreicht?“

## ■ Ferien: Nimm dir Zeit für dich!

## ■ Wieder zur Uni

- Stundenplan überdenken
- Stoff wieder einarbeiten
- Prüfungsvorbereitung beginnen

## Reflexion Ressourcen

- Zentrale Studienberatung ([ZSB](#))
- Angebote der Fakultät für Informatik
  - Informatik Studiengangservice ([ISS](#))
  - [eezi-Beratungs Gespräch](#)
  - Fachschaft ([FSMI](#)) / [Wi-Forum](#)
- Mit Studierenden oder Tutoren aus höheren Fachsemestern reden
- Mit sozialem Umkreis und Familie reden

# EURE FRAGEN, UNSERE ANTWORTEN!

# DANKE FÜR EURE AUFMERKSAMKEIT!

## Orientierungsprüfungsvorbereitung

- Themen
  - Vorbereitungsstrategie für Prüfungen
  - Fachliche Tipps zu Orientierungsprüfung
  - Anschließende Lernpartnerschaftsbörse / Studygroups für Prüfungsauserephase
- Am **21.01.25**, um **17:30 Uhr**
- [20.40 Fritz-Haller-Hörsaal](#) (HS37)  
und auf Zoom



# Beratung und Auskunft

KIT-Fakultät für Informatik

Startseite > Studium > Beratung & Information > Beratung

## Beratung und Auskunft



### Angebote der Fakultät für Informatik

- ISS - Informatik Studiengangservice: Studienberatung, Verlängerungsgarantie, Studienplänenänderung, usw.
- KIT-Informatik-Fakultätsbibliothek: Lernplätze, Literatur, usw.
- ATIS - Abteilung Technische Infrastruktur: E-Mail-Dienste, Applikation, Multimediale Räume, Software-Bereitstellung

### Studentische Beratung

- Fachschaft Mathematik / Informatik [GF](#): Beratung durch Studierende des Faches Informatik / Mathematik
- Forum Wirtschaftsinformatik [GF](#): Beratung durch Studierende der Fachrichtung Wirtschaftsinformatik

### Zentrale Angebote des KIT

- ZSB - Zentrale Studienberatung: Zweifel am Studium, Studiengang, -Hochschulwechsel, Studienabbruch, usw.
- SCC - Scientific Computing Center: Hilfestelle für technische Probleme, Accountregistrierung, usw.
- MINT-Kolleg: Vorkurse, Semesterkurse, Helpdesk, Aufbaukurse
- HoC - House of Competence: Lern-/Schreibberatung, Helpdesk
- ZLB - Zentrum für Lehrerbildung: Ressourcen für Lehramt-Studierende, usw.
- ZML - Zentrum für Mediales Lernen: Lehrveranstaltungen
- KIT-Bibliothek: Lernplätze, Online-Katalog, usw.
- Hochschulspott.am.KIT

### Studierendenwerk Karlsruhe

Das Studierendenwerk Karlsruhe [GF](#) unterstützt bei Themen wie:

- BAföG, Stipendien & Darlehensberatung
- Wohnheimplätze und Zimmervermittlung
- Sozial-, Rechts- & Psychotherapeutische Beratung
- Studieren mit Kind

### Angebote für internationale Studierende

- ISIO - International Students Office: Beratung, Betreuung, Begleitung, Integrationsmaßnahmen
- International Student Center [GF](#): Sozialberatung, Ausländische Studierende, Studieren mit Behinderung, Wohnungssuche, usw.
- Studienstart International: Willkommensveranstaltungen, Informationsangebote, Campusprechstunden, usw.
- Studienstart International - Infoangebot auf ILIAS
- Forschungsaufenthalt am KIT: Mitwirken an Praktika, Forschungsprojekten und Abschlussarbeiten für internationale Studierende
- ESN - Erasmus Students Network Karlsruhe [GF](#): Für sowohl internationale Studenten als auch incoming + Outgoing Austauschstudenten
- Koordinationsstelle für Geflüchtete und MigrantInnen: Persönliche Sprechstunden, Informationsabende, Integrationsmaßnahmen
- Ausländerbehörde der Stadt Karlsruhe: [GF](#): Aufenthaltslaubnis, Elektronischer Aufenthaltstitel (eAT), Niederlassungserlaubnis, usw.
- Das ABC des KIT: Englisch-Glossar zur Übersetzung von fachspezifischen Begriffen
- Buddy-Programme: für Abschluss-suchende und Exchange-Studenten
- MINTership Inclusive: [GF](#) YouTube-Video über Auslandsaufenthalte für Studierende mit Behinderung
- International Student Hub: Fragen zum Studium und zum Alltag, Unterstützung bei Schwierigkeiten und Auskunft über die Finanzierung des Studiums und über Freizeitaktivitäten

### Weitere Beratungsstellen

- AStA - Allgemeiner Studierendenausschuss [GF](#): Sozialberatung, Chancengleichheit, Querberatung
- PBS - Psychotherapeutische Beratungsstelle für Studierende [GF](#): Stress, Prüfungsangst, Einsamkeit
- BBC - Beauftragte für Studierende mit Behinderung und chronischer Krankheit
- ACCESSKIT - Zentrum für digitale Barrierefreiheit und Assistive Technologie
- Career-Service: Berufsinstieg, Praktikumsuche, Karriereberatung
- Telefonseelsorge [GF](#): Zuhörtelefon bei persönlichen Problemen
- Nightline Karlsruhe [GF](#): Zuhörtelefon von Studierenden bei persönlichen Problemen