

# Algorithmen und Datenstrukturen

## Sommersemester 2022

Einführung

Fabian Kuhn

Algorithmen und Komplexität



**UNI  
FREIBURG**

- Letztes Semester haben Sie (die Informatiker/innen) die Grundzüge des Programmierens gelernt
- Fragen der Effizienz spielten eine untergeordnete Rolle
- Dies ist das generelle Thema dieser Vorlesung
  - **Wie schnell ist mein Programm? Wie misst man das?**
  - **Wie kann ich es schneller machen?**
  - **Wie kann ich beweisen, dass es immer so schnell läuft?**
- Manchmal geht es auch um Sparsamkeit im Platzverbrauch oder andere Ressourcen, aber hier meistens um **Zeit**

- Sie können schon **kleinere Programme schreiben**
  - z. B.: die kleinste Zahl in einem Array finden, einen gegebenen einfachen Algorithmus implementieren, ...
- Sie haben **Erfahrung in einer Programmiersprache** gesammelt
  - Offiziell werden wir in dieser Vorlesung **Python** unterstützen
- Verständnis von **wichtigen Grundkonzepten der Programmierung**
  - Variablen, Objekte, Schleifen, Ein- und Ausgabe, Funktionen, Rekursion
- **Falls Sie da Lücken haben**, müssen Sie das parallel zur Vorlesung aufarbeiten und haben dann **deutlich mehr Aufwand!**

## Algorithmen:

- Algorithmenentwurf: Wie löst man ein geg. Problem effizient?
- Wie analysiert man einen Algorithmus?
  - O-Notation, Laufzeitanalyse, Korrektheit (auch math. Beweise)
  - Sie sollten z. B. nach der Vorlesung verstehen, ob Ihr Programm
    - in Linearzeit oder “fast-Linearzeit” läuft (gut!)
    - quadratische oder höhere polynomielle Laufzeit hat (oft schlecht!)
    - exponentielle Laufzeit hat (immer schlecht!)
- Beispiele: Sortieren, Suchen, Graphenalgorithmen (kürzeste Wege, Spannbäume, Breiten-/Tiefen-Suche), Editierdistanz

## Datenstrukturen:

- Wie legt man Daten ab, so dass man schnell darauf zugreifen kann?
- Gute/geeignete Datenstrukturen  $\Leftrightarrow$  effiziente Algorithmen
  - Wir müssen gute Algorithmen haben, um effizient auf Daten zuzugreifen
  - Daten müssen geschickt abgelegt werden, damit effiziente Alg. Existieren
  - Die Laufzeit von Algorithmen für grössere Probleme hängt oft eng damit zusammen, welche Datenstrukturen man verwendet, um die anfallenden Daten zu verwalten
- Beispiele: Hashtabellen, Suchbäume, (Prioritäts-)Warteschlangen, dynamische Arrays, Repräsentation von Graphen

## Vorlesung: Inverted Classroom

- Wir stellen Videos mit den Vorlesungsinhalten zur Verfügung
  - 12 Videos, jeweils am Anfang der Woche, 1. Video diese Woche
- Wöchentliche Veranstaltung zur Diskussion der Inhalte
  - Jeweils Dienstag, 10:15 – 12:00
  - Präsenzveranstaltung, Teilnahme via Zoom möglich
    - Wir beabsichtigen **nicht**, diese Veranstaltungen aufzuzeichnen
    - Teilnehmende via Zoom werden allenfalls nicht alle Diskussionen im Vorlesungssaal verfolgen können
      - Solange es nicht zu viel ist, werden wir versuchen, Fragen aus dem Vorlesungssaal für die Teilnehmenden im Zoom-Meeting zu wiederholen.
  - Beantworten / diskutieren von Fragen zu Vorlesung / Übungen
  - Vertiefung von wichtigen Aspekten

**Webpage:** [http://ac.informatik.uni-freiburg.de/teaching/ss\\_22/ad-lecture.php](http://ac.informatik.uni-freiburg.de/teaching/ss_22/ad-lecture.php)

## Übungsblätter

- Ein Übungsblatt pro Woche (12 Übungsblätter)
  - Übungen ca.  $\frac{1}{2}$  theoretisch und  $\frac{1}{2}$  praktisch
  - pro Übungsblatt sind 20 Punkte möglich
- Ausgabe der Übungen ist jeweils am Dienstag
- **Abgabe der Übungen** ist bis **Dienstag um 10:00** in der Folgewoche
  - Abgabe der Übungen ist online via Daphne (Details folgen)
- Um die Studienleistung zur Vorlesung zu bestehen, müssen Sie **50% der Punkte** (Total: 120 Punkte) aus den Übungen machen.

## Online-Übungsbetrieb / Assistenten und Tutoren

- Keine regelmässigen Übungsgruppentermine (alles online)
- Assistenten: [Philipp Bamberger](#), [Marc Fuchs](#), [Philipp Schneider](#)
- Tutoren: Tjark Behrens, Hendrik Jansen, Nelson Paraiso, Florian Pollitt, Gustav Schmid, Matthias Zumkeller

- Die Übungsblätter können in **Gruppen** bestehend aus **maximal 3 Personen** bearbeitet werden
  - Wir empfehlen, die Übungen in Gruppen zu bearbeiten
  - Sorgen Sie dafür, dass alle in einer Gruppe zur Lösung beitragen und somit von den Übungen profitieren.
- Bitte bilden Sie feste Gruppen, um die Übungen zu bearbeiten
  - Bitte die gebildeten Übungsgruppen per E-Mail an Philipp Bamberger ([philipp.bamberger@uni-freiburg.de](mailto:philipp.bamberger@uni-freiburg.de)) melden.
  - Wir werden über das Zulip-Forum auch noch eine Möglichkeit zur Verfügung stellen, um Gruppen zu bilden
- Nur eine Übungsabgabe pro Gruppe
  - Jede/jeder muss eine Lösung abgeben. Mitglieder einer Gruppe sollen aber die gleiche Lösung abgeben
    - Die Tutoren werden nur eine Lösung pro Gruppe korrigieren und dann allen Mitgliedern der Gruppe das gleiche Feedback zurückschreiben.

## Die Übungen sind der wichtigste Teil der Vorlesung!

- Das Lösen der Übungen ist grundsätzlich die beste Klausurvorbereitung!
- Wenn Sie die Übungen alle sorgfältig gemacht und verstanden haben, sollte die Klausur gut machbar sein.

## Selbstständiges Erarbeiten der Lösungen

- Sie müssen die Lösungen selbst (in Ihren Gruppen) erarbeiten.
- Kopien bestehender Lösungen / Plagiate sind nicht erlaubt.
  - Sie dürfen natürlich Ideen mit Ihren Kolleg:innen, bzw. über's Forum diskutieren und auch im Internet recherchieren...

## Aufwand

- 6 ECTS = ca. 180 Arbeitsstunden
- Bei 120h Vorlesungen/Übungen und 60h Klausurvorbereitung
  - im Schnitt ca. **6-7 Stunden pro Übung...**

## benutzen **Daphne** als **Kursverwaltungssystem**

- Link auf der Vorlesungswebpage: **bitte anmelden!**
- In Daphne haben Sie eine Übersicht über folgende Infos
  - Wer ihr/e Tutor/in ist
  - Ihre Punkte in den Übungsblättern
  - Infos zum aktuellen Übungsblatt
  - Link zum **SVN**
  - Link zu den **Coding Standards**
  - Link zum **Build System**
- Das gleiche System wird auch in anderen Vorlesung verwendet

SVN : <http://subversion.apache.org>

- Kennen Sie bereits aus Informatik I?
- Dateien liegen auf einem zentralen Server, in einem sogenannten **repository**, die typische Operationen sind
  - **Update**: neuste Version vom Server ziehen
  - **Commit**: letzte Änderungen auf den Server hochladen
- Vollständige Historie von allen Änderungen an den Dateien
- Insbesondere nützlich für das Schreiben von Code
- Wir werden dies benutzen für
  - die **Abgaben Ihrer Übungsblätter** (Code + alles andere)
  - das **Feedback von Ihrem Tutor**
  - Vorlesungsdateien / Musterlösungen

## Richtlinien für die Abgabe der Programmierübungen:

- Für die Übungen verwenden wir Python als Programmiersprache
- **Unit tests** für alle nichttrivialen Methoden
  - Ohne Test ist die Chance gross, dass die Methode falsch ist...
  - Immer **mind. eine typische Eingabe** und **einen kritischen Grenzfall** testen!
  - Vereinfacht das Debuggen von grösseren Projekten
  - Wir/Sie können so testen, ob das Programm das Richtige tut
- Befolgen eines **einheitlichen Programmierstils** (style checker)
  - Macht den Code leserlich (auch für andere!)
- Standardisiertes **Build-Framework** (jenkins)
  - So können wir effizient Ihre Programme ausprobieren und testen

- Wir benutzen zusätzlich Zulip als Online-**Forum** für Fragen, Diskussionen, weitere Informationen, etc.
  - Zulip ist ein Gruppen-Chat/-Forum (<https://zulip.com>).
  - Verwenden Sie Zulip für Fragen zur Vorlesung und zu den Übungen.
  - Wir werden Zulip auch für zusätzliche Informationen verwenden.
- Erklärungen zum Zugang kommen noch...
- Falls Sie Fragen zur Vorlesung oder den Übungen haben, benutzen Sie **Zulip anstatt eine E-Mail** an uns zu schreiben!
  - So sehen Ihre Kolleg\*innen die Fragen auch und können sich auch an der Diskussion beteiligen.
  - Wir können Fragen direkt für alle beantworten
  - Sie können Zulip gerne für alle Fragen und Diskussionen benutzen, die mit dem Stoff oder der Organisation der Vorlesung zu tun haben.

## Zulip hat eine 2-Level Hierarchie

- 1. Level: Streams
  - Die public Streams sind von uns vordefiniert
  - Sie können private Streams selbst definieren, um z.B. innerhalb Ihrer Übungsgruppe zu kommunizieren
- 2. Level: Topics
  - Jede Nachricht hat ein Topic. Nachrichten können nach ihrem Topic gruppiert werden. Bitte benutzen Sie kurze, aussagenkräftige Topics, wenn Sie neue Topics definieren.

## Zulip Streams für die Vorlesung:

- **Allgemein\_SS22** : Fragen / Ankündigungen bezügl. Organisation
- **Vorlesung\_SS22** : Fragen / Diskussionen zu den Vorlesungen
- **Übung\_SS22** : Fragen / Diskussionen zu den Übungen
- **Technik\_SS22** : Fragen / Diskussionen zu SVN, Daphne, etc.

# Zugang zu Zulip und Daphne

Infos zur Anmeldung sind auf der Vorlesungs-Webseite

- Sie müssen sich für beide Systeme separate anmelden!

## Zulip

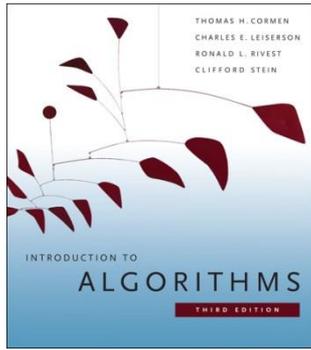
- Sign-up Link ist auf der gleichen Seite wie der Zoom Link
  - Nur innerhalb des Universitäts-Netzwerks zugänglich (z.B., über VPN)

## Daphne

- Sign-up Link ist auf der Haupt-Vorlesungs-Webseite
- Sie brauchen Ihr RZ-Login, um sich für Daphne anzumelden!

[http://ac.informatik.uni-freiburg.de/teaching/ss\\_20/ad-lecture.php](http://ac.informatik.uni-freiburg.de/teaching/ss_20/ad-lecture.php)

- Alle wichtigen Informationen zur Vorlesung
- Vorlesungsvideos, Übungsblätter, Musterlösungen, etc.
- Link zu Daphne, Zulip, etc.
- Erklärungen zum Umgang mit Daphne
  - Wir werden gleich noch eine kurze Einführung geben
- Sonstige wichtige Informationen
- ...



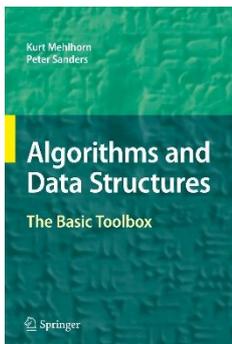
T. Cormen, C. Leiserson, R. Rivest, C. Stein  
Introduction to Algorithms, Third Edition,  
MIT Press, 2009

## **Klassisches Lehrbuch zum Thema (auf Englisch)**



T. Ottmann, P. Widmayer  
Algorithmen und Datenstrukturen, 5. Auflage,  
Spektrum Akademischer Verlag, Heidelberg, 2012

## **Deutsche Alternative (aus Freiburg / Zürich)**



K. Mehlhorn, P. Sanders  
Algorithms and Data Structures,  
Springer, 2008, online verfügbar unter

<http://www.mpi-inf.mpg.de/~mehlhorn/Toolbox.html>

## Wikipedia:

- ist bei Standardalgorithmen / -datenstrukturen recht gut...
- z.T. auch für weitergehende Algorithmen

## Alte Vorlesungen, z.B.:

- Material von früheren Informatik II / Alg. & Datenstr. Vorlesungen
  - 2019: <https://ad-wiki.informatik.uni-freiburg.de/teaching/AlgoDatSS2019>
  - 2018: [http://ac.informatik.uni-freiburg.de/teaching/ss\\_18/info2.php](http://ac.informatik.uni-freiburg.de/teaching/ss_18/info2.php)
  - ...
- Vorlesungen von anderen Unis, z.B.:  
MIT Courseware: <http://ocw.mit.edu>
  - u.a. mit Aufzeichnungen von 2011