Algorithmen und Datenstrukturen

– Wintersemester 2019 –

Organisatorisches

Fachbereich Informatik
TU Kaiserslautern

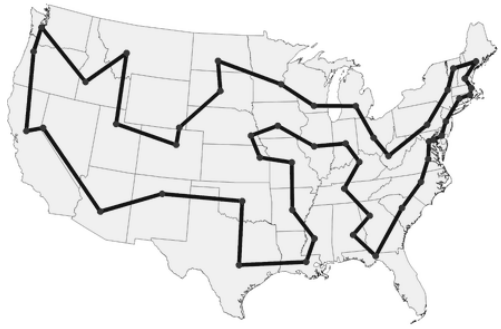
Dozent: Dr. Patrick Michel

Folien ursprünglich von Prof. Dr. Adrian Ulges (Hochschule RheinMain)

Warum "Algorithmen und Datenstrukturen"?

In der Informatik gibt es viele **wiederkehrende Kernprobleme**:

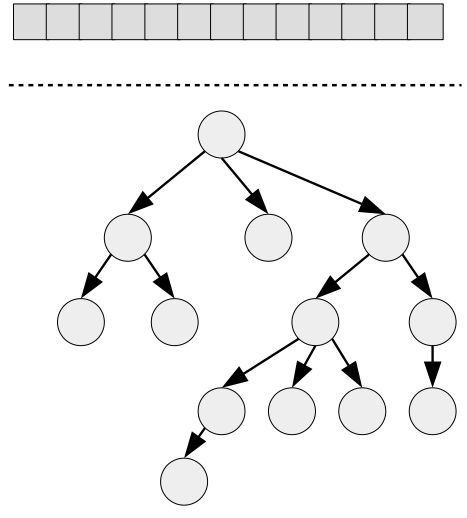
- Große Datenmengen abspeichern
- in Datenmengen suchen
- Daten sortieren
- Problemlösungen finden



Ziele von ADS

In ADS wollen wir diese **zentralen Probleme besser verstehen**:

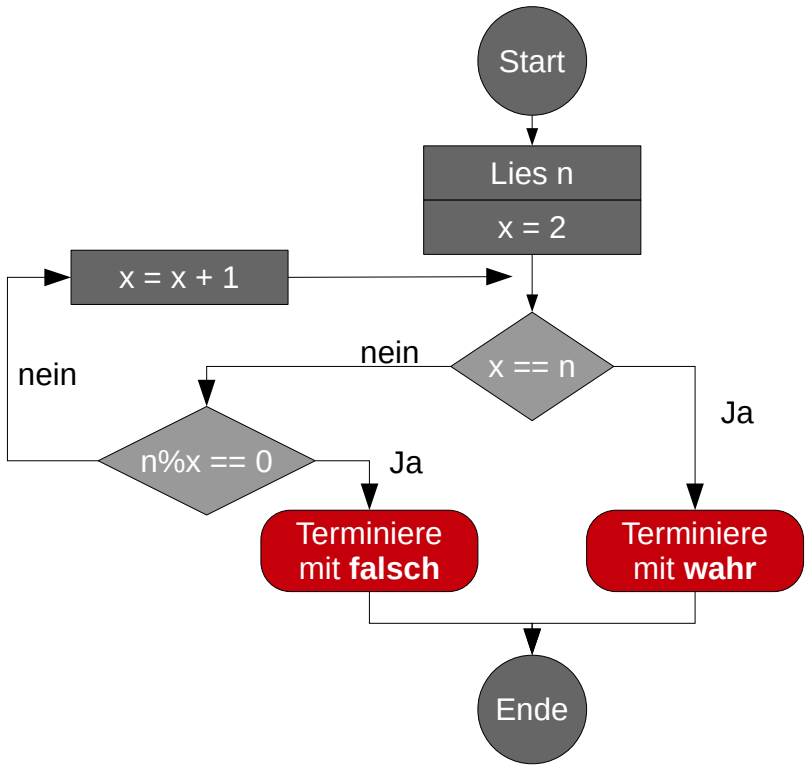
- **Lösungen entwickeln** *(unabhängig von der jeweiligen Programmiersprache und Domäne)*.
- "Gute" Standardlösungen kennen: **Handwerkszeug des Informatikers** *(Quicksort, Hashing, B-Bäume, ...)*.
- Verstehen was "gut" / "effizient" bedeutet **(Komplexität)**.
- Praktische Realisierung in **Java**.



Algorithmen-Grundlagen

Wir beschäftigen uns mit dem **Algorithmenbegriff**:

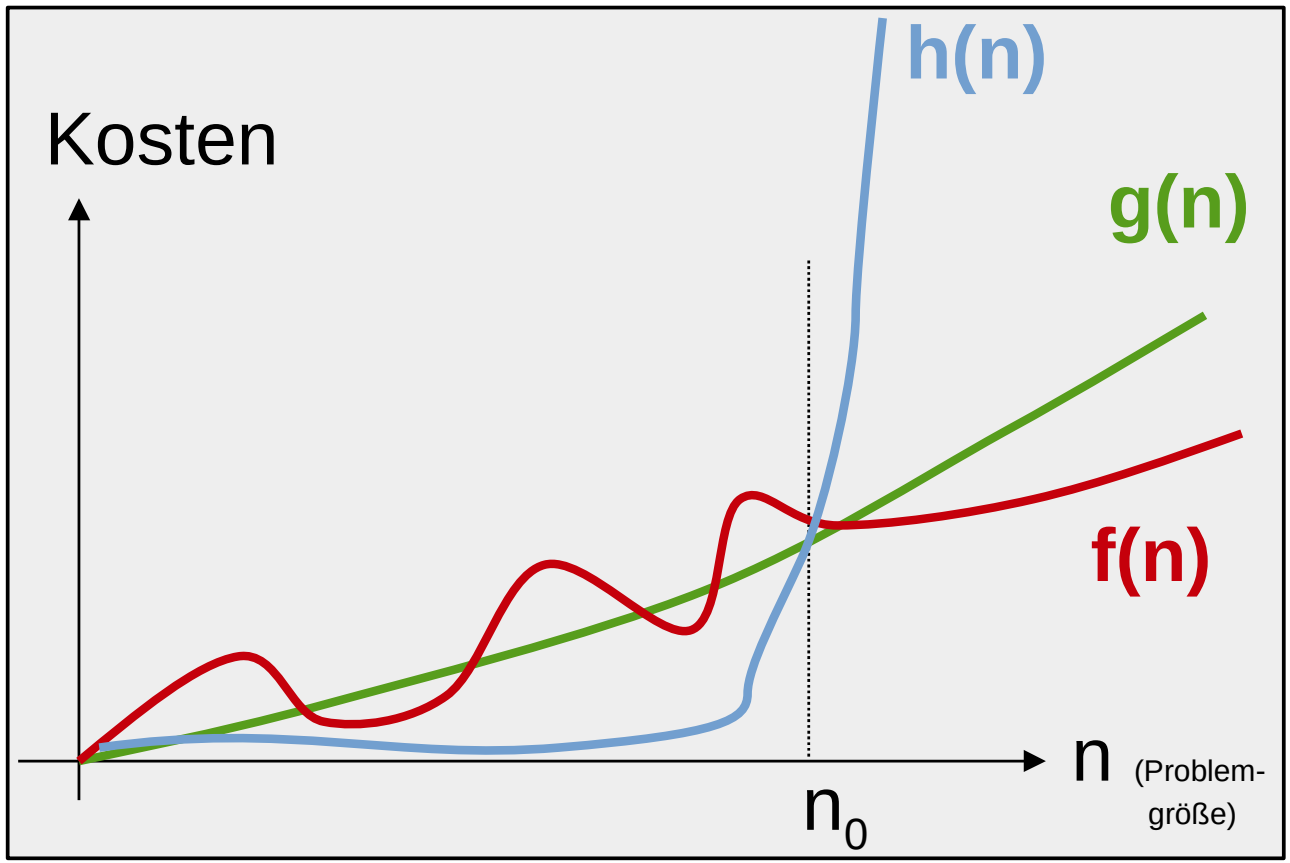
- Definition, Historie
- Wichtige Eigenschaften von Algorithmen
- Darstellung von Algorithmen



Komplexität

Wir beschäftigen uns mit der **Effizienz** von Algorithmen:

- Laufzeit berechnen
- Skalierbarkeit abschätzen: Die O-Notation.



Sortieren

Wir beschäftigen uns mit dem Kernproblem **“Sortieren”**:

- Standard-Algorithmen *(Quicksort, Mergesort, Heapsort, ...)*.
- Externes Sortieren riesiger Datenmengen.
- Was ist der effizienteste Sortieralgorithmus?

```
void sort1(int[] a) {
  int n = a.length;
  int newn;
  do {
    newn = 1;
    for (int pos=0; pos < n-1; p
      if (a[pos] > a[pos+1]) {
        swap(a, pos, pos+1);
        newn = pos+1;
      }
    }
    n = newn;
  } while (n>1);
}
```

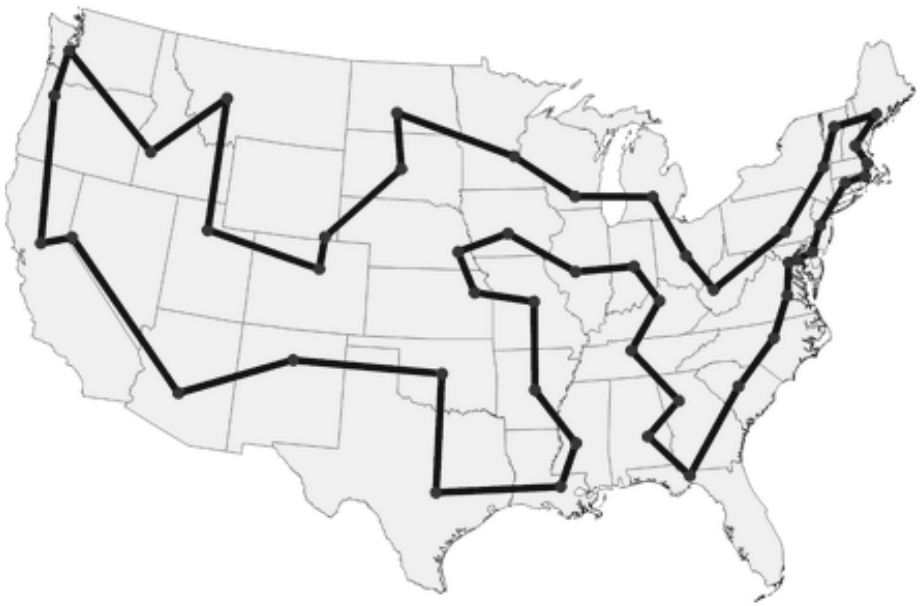
VS.

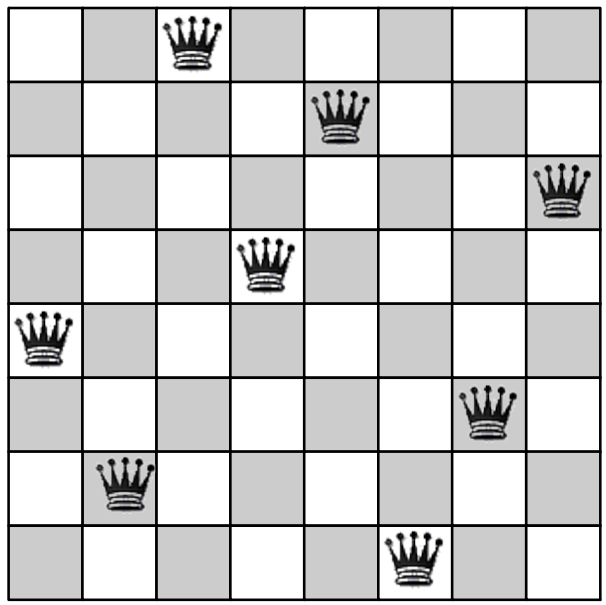
```
void sort2(int[] a) {
  for (int pos=0; pos<a.length; pos++) {
    int min = pos;
    for (int minkand=pos;
         minkand < a.length;
         minkand++) {
      if (a[mindkand] < a[min]) {
        min = minkand;
      }
    }
    swap(a, pos, min);
  }
}
```

Algorithmenmuster

Wir betrachten das Lösen von Daten-Problemen:

- Verschiedene Problemtypen *(Packen, Planen, Anordnen, ...)*
- **Schemata** von Lösungsstrategien.
- Beurteilung ihrer Güte / Optimalität.

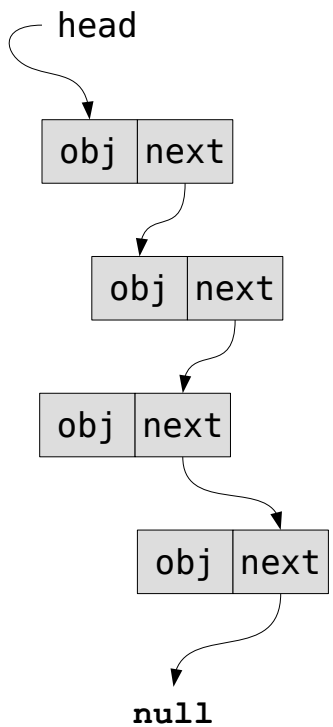




Abstrakte Datentypen + Verkettete Listen

Wie organisieren wir Daten **effizienter als mit Arrays**?

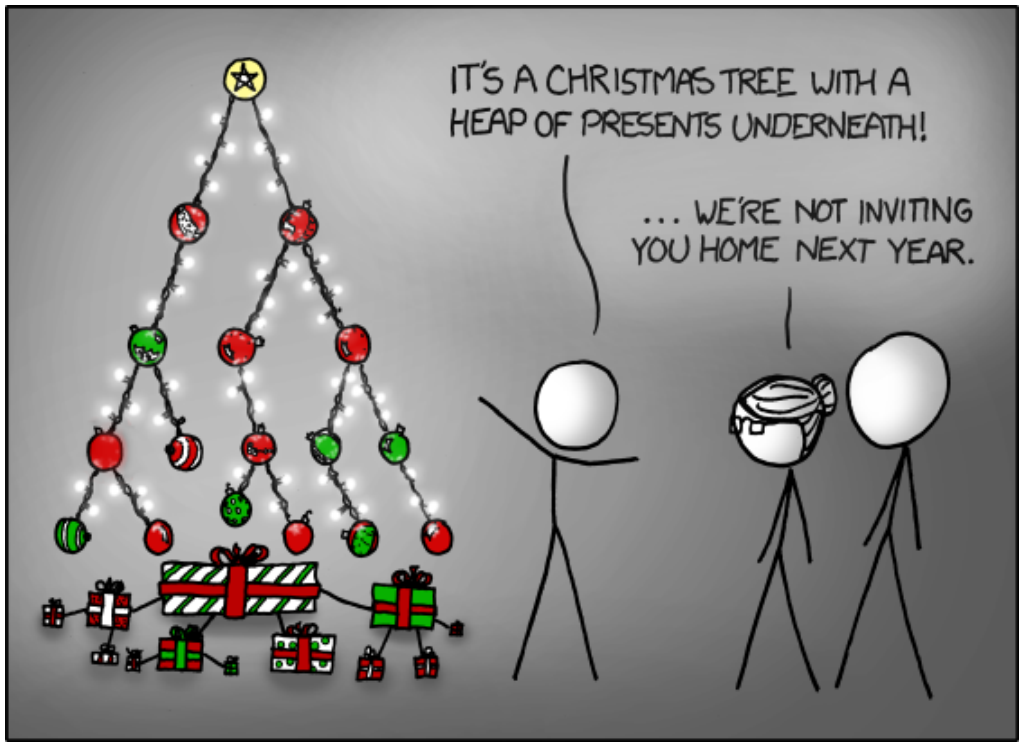
- **Datentypen**: Stack, Queue, Priority Queue, ...
- Implementierung **dynamischer Kollektionen** durch Referenzen.



Bäume

Bäume = **Die Datenstruktur** der Informatik

- **Definitionen** *(Vollständigkeit, Wurzel, Höhe, ...)*
- Suchbäume, balancierte Bäume
- **Datenbanken**: B-Bäume



Hashing

Effizientes Speichern von Kollektionen mit **Hashing**

- Grundidee: **Fingerprinting** mit möglichst guter **Streuung**.
- Eigenschaften "guter" Hash-Funktionen.
- Behandlung von Kollisionen.

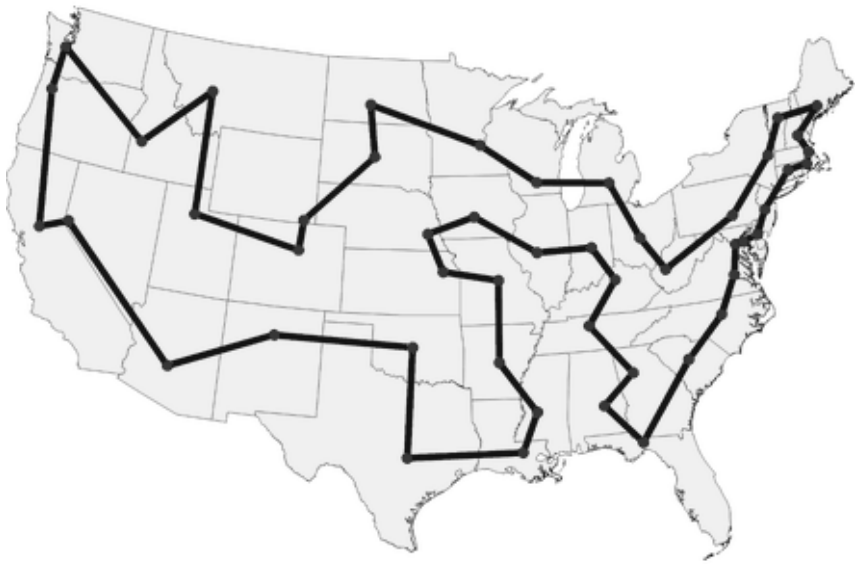
=

```
79054025
255fb1a2
6e4bc422
aef54eb4
```

Graphen

Wahrscheinlich **die allgemeinste** Datentrukturen der Informatik.

- Verallgemeinerung von Bäumen.
- Sehr praxisrelevant.
- Standardprobleme: Kürzeste Wege, Travelling Salesman, etc.



Grundlagen der Veranstaltung

Programmieren

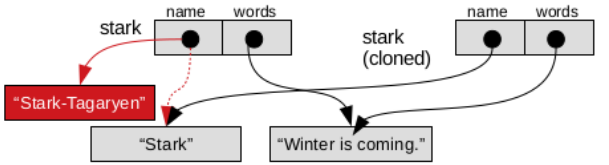
- Flüssiges Coden in Java
- Grundkonzepte: Iteration, Rekursion
- Referenzierung.

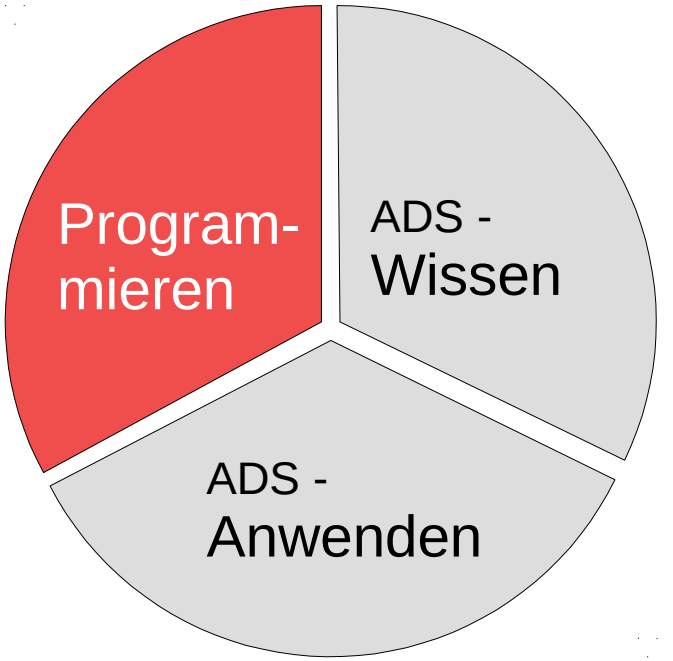
Anmerkungen

- Erwarten Sie in **Klausur** und **Übungen** eine Mischung aus
  (a) Programmieren
  (b) "ADS-Wissen"
  (c) Anwenden von ADS-Verfahren.

```java
public class Character {
    String name;
    House house;

    public Character(String name,
                     House house) {
        this.name = name;
        this.house = house;
    }
```





Anlaufpunkte

Webseite

- Zentraler Anlaufpunkt für die Veranstaltung
- `softech.cs.uni-kl.de`
- Unter Lehre: Vorlesung "Algorithmen und Datenstrukturen"
- Treten Sie der Veranstaltung in **Exclaim** bei und tragen Sie sich dort in die Übung ein!

Dozent

- Dr. Patrick Michel, **Mail**: uni@pbmichel.de
- Jederzeit ansprechen!

Tutorin

- Danielle Korth, **Mail**: dkorth@rhrk.uni-kl.de
- Jederzeit ansprechen!

Genereller wöchentlicher Ablauf

- **DO abend** (wenn möglich **DI abend**): Slides + Übungsblatt auf Webseite.
- **DO abend**: Vorlesung.
  – Übungsblatt bearbeiten –
- **MO abend**: Abgabe der Übung (Theorie + Praxis).
  Es werden i.d.R. keine Musterlösungen zur Verfügung gestellt.
  - Theorie in Übungskasten (4. Stock, zwischen Geb. 32+34)
  - Praxis in Exclaim hochladen
- Praxis **bis MI** verbessern
- **MI morgen**: Gemeinsame Übungsgruppe
  – Finale Abnahme durch Tutor –

- Bei Problemen nicht zögern Hilfe zu suchen!

Wie bestehe ich dieses Modul?

Um das Modul zu bestehen, müssen Sie zur Klausur zugelassen werden, diese schreiben und bestehen.

Zulassung

- Alle bis auf maximal zwei Übungsblätter **sinnvoll bearbeitet**.
- Abgabe in Gruppen, Punkte werden aber **individuell** geführt.
- Achten Sie darauf, dass **Sie selbst** Lösungen programmieren und herleiten können und den Stoff verstehen.
  - **Jeder** muss **selbst** programmieren um es wirklich zu lernen.

Klausur

- Mischung aus Programmieren, ADS Wissen und Anwenden von ADS Verfahren.
- Wer sich **selbst** mit dem Stoff beschäftigt, die Übungen **selbst** gelöst und viel **selbst** programmiert hat, wird sie gut schaffen.