

# Grundbegriffe der Theoretischen Informatik

Sommersemester 2018 - Thomas Schwentick

## 0: Einleitung

Version von: 10. April 2018 (11:40)

# Inhalt

## ▷ 0.1 Was ist Theoretische Informatik?

0.2 Themen der GTI-Vorlesung

0.3 Literatur

0.4 Organisatorisches

# Warum Theorie der Informatik helfen kann


- 2013 in der Presse

 heise.de

- Um auf die Kontaktliste eines gesperrten iPhones zuzugreifen, konnte man
  - es einschalten
  - den „Entsperren“-Slider nach rechts ziehen
  - auf „Notruf“ tippen
  - auf den Ausschalt-Knopf drücken, bis die Ausschalt-Option angezeigt wird
  - „Abbrechen“ klicken
  - eine Notrufnummer wählen, verbinden  
und sofort wieder auflegen
  - das Gerät aus- und einschalten
  - den „Entsperren“-Slider ziehen
  - den Ausschaltknopf gedrückt halten
  - kurz bevor die Ausschalt-Option angezeigt wird, auf „Notruf“ drücken
- Die Kontakte öffneten sich und blieben geöffnet, solange der Ausschaltknopf gedrückt bleibt
- Besser wäre es, automatisch überprüfen zu können, ob es eine solche Lücke gibt...

# Wichtige Themen der Informatik – sehr grob



- Schritte auf dem Weg von der Programmidee zum Programm
  - ... und zugehörige Lehrveranstaltungen  ganz grob
- Spezifikation:
  - Software-Technik, DAP I, Formale Methoden des System-Entwurfs
- Wahl geeigneter Algorithmen:
  - DAP II, Effiziente Algorithmen, Komplexitätstheorie
- Umsetzung der Algorithmen in ein Programm:
  - DAP I, Software-Technik, SoPra
- Übersetzung des Programms:
  - Übersetzerbau
- Ausführung des Programms:
  - Rechnerstrukturen, Betriebssysteme, HaPra
- Die GTI bezieht sich auf mehrere dieser Schritte

# Wunschzettel der Informatiker

- Wünsche für die einfache Softwareerstellung
  - Zu jeder Spezifikation sollte es einen Algorithmus und ein Programm geben
  - Programme sollten aus der Spezifikation automatisch erzeugt werden
  - Es sollte automatisch überprüft werden können, ob ein Programm seiner Spezifikation entspricht
  - Programme sollten möglichst klein sein
  - Programme sollten möglichst schnell sein
  - Die Übersetzung von Programmen in Maschinenprogramme sollte automatisch erfolgen
- In dieser Vorlesung werden wir Grenzen für die Erfüllung dieser Wünsche kennen lernen
- Diese Grenzen gehören zu den ersten Erkenntnissen der Theoretischen Informatik — bevor es mit der praktischen Informatik so richtig los ging
- Aber was ist überhaupt Theoretische Informatik?

# Was ist Theoretische Informatik? (1/2)

- Was ist Theoretische Informatik?
- Die Antwort auf diese Frage hat zwei Komponenten:
  - Die Methoden der Theoretischen Informatik
  - Die Inhalte der Theoretischen Informatik
- Die Theoretische Informatik verwendet die **mathematische Methode** für die **Grundlagen der Informatik**
- Ganz grob lässt sich das durch die folgende „Gleichung“ ausdrücken:

$$\frac{\text{Theoretische Informatik}}{\text{Informatik}} \approx \frac{\text{Mathematik}}{\text{Physik}}$$

# Methodik der Theoretischen Informatik

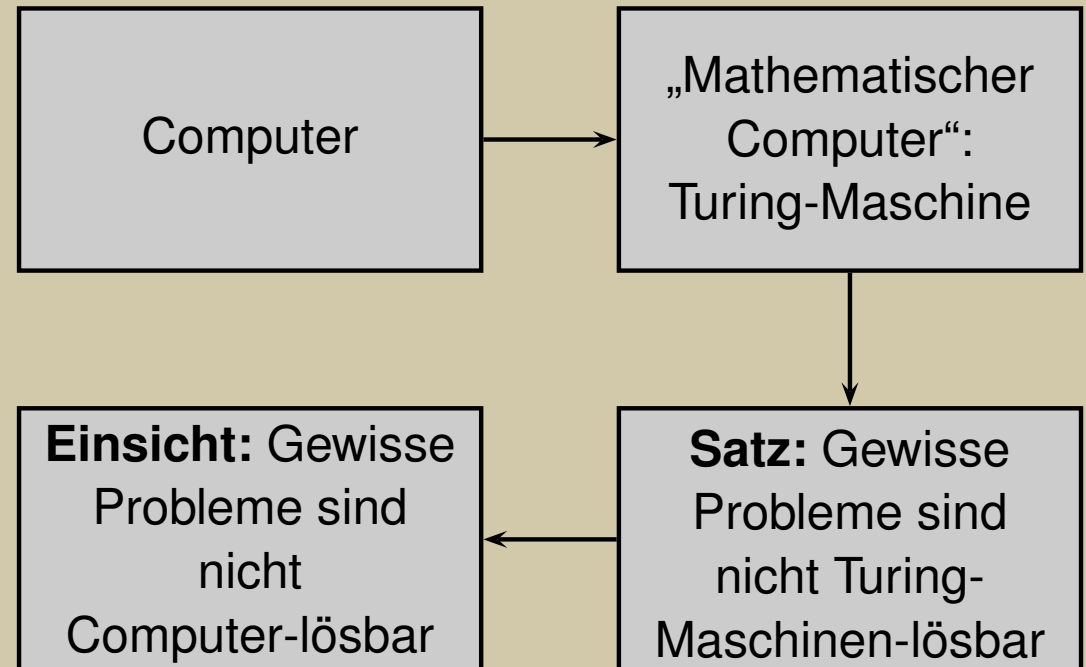
- **Typische Vorgehensweise:**

- Modellierung von im Zusammenhang mit Computern und Berechnungen auftretenden Phänomenen durch mathematische Objekte
- Formulierung von Aussagen über diese Objekte
- Beweis dieser Aussagen

- Warum ist der mathematisch präzise Ansatz (inklusive Beweisen) für die Informatik sinnvoll?

- Die Semantik von Spezifikationen sollte präzise definierbar und beispielsweise die Äquivalenz von Spezifikation und Modell beweisbar sein
- Auch die Semantik von Algorithmen/Programmen sollte präzise definierbar sein und Korrektheits- und Aufwandsaussagen sollten sich beweisen lassen

## Beispiel



# Was ist Theoretische Informatik? (2/2)

Artikel

Diskussion

Lesen

Bearbeiten

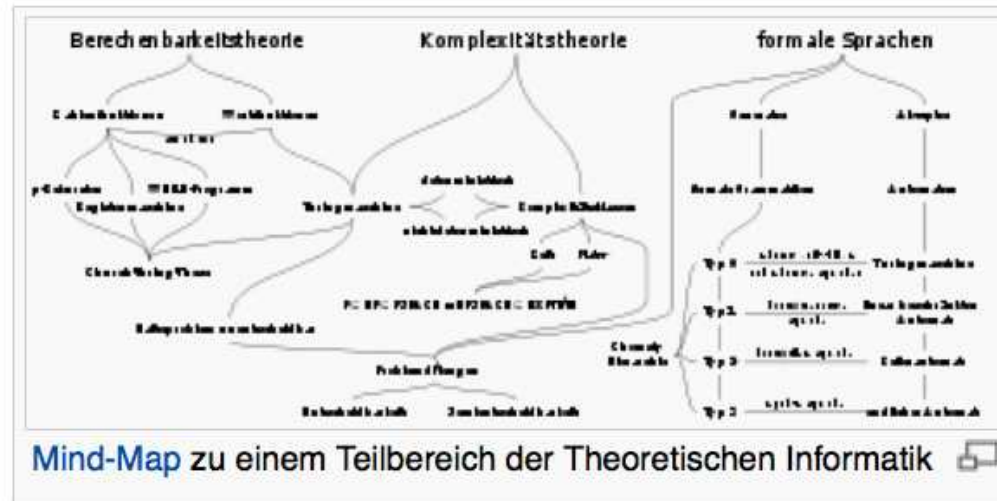
Suche



## Theoretische Informatik

Die **Theoretische Informatik** beschäftigt sich mit der Abstraktion, Modellbildung und grundlegenden Fragestellungen, die mit der Struktur, Verarbeitung, Übertragung und Wiedergabe von **Informationen** in

Zusammenhang stehen. Ihre Inhalte sind **Automatentheorie**, Theorie der **formalen Sprachen**, **Berechenbarkeits-** und **Komplexitätstheorie**, aber auch **Logik** und **formale Semantik** sowie die **Informations-**, **Algorithmen-** und **Datenbanktheorie**.



©Wikipedia



# Teilgebiete der Theoretischen Informatik (1/3)

- **Berechenbarkeit: „Was ist überhaupt möglich?“**
  - Was ist ein Algorithmus?
  - Welche Probleme lassen sich algorithmisch lösen?
  - Wie lassen sich die algorithmisch nicht lösbaren Probleme klassifizieren?
- **Komplexitätstheorie: „Was ist effizient möglich?“**
  - Klassifikation algorithmischer Probleme nach ihrer prinzipiellen Schwierigkeit
  - Vergleich der dabei entstehenden Klassen
  - Besondere Berechnungsmodi: zufallsgesteuert, parallel, nicht-deterministisch
  - **P** vs. **NP**
- **Effiziente Algorithmen: „Wie geht es am schnellsten?“**
  - Effiziente Algorithmen für konkrete Probleme
  - Datenstrukturen
  - Algorithmische Geometrie, Graph-Algorithmen, ...
  - Parallele Algorithmen, Online-Algorithmen, ...

## Teilgebiete der Theoretischen Informatik (2/3)

- **Formale Sprachen: „Wie sieht ein Programm aus?“**
  - Wie lässt sich die syntaktische Struktur von Computer-Programmen beschreiben?
  - Reguläre Sprachen
  - Kontextfreie Sprachen
  - Sprachen von unendlichen Zeichenketten, Bäumen etc.
- **Semantik: „Was bedeutet ein Programm?“**
  - Wie lässt sich die Bedeutung/Wirkung von Computer-Programmen so formalisieren, dass z.B. ihre Korrektheit bewiesen werden kann?
  - Semantik von Programmiersprachen
  - Funktionale und logische Programmiersprachen
  - Model Checking
- **Viele weitere Gebiete:**
  - Theorie verteilter Systeme
  - Datenbanktheorie
  - ...

## Teilgebiete der Theoretischen Informatik (3/3)

- Diese Aufzählung der Teilgebiete der Theoretischen Informatik ist nicht vollständig
- Die **Special Interest Group on Algorithms and Computation Theory (SIGACT)** der **Association for Computing Machinery (ACM)** schreibt dazu:
  - „TCS covers a wide variety of topics including algorithms, data structures, computational complexity, parallel and distributed computation, probabilistic computation, quantum computation, automata theory, information theory, cryptography, program semantics and verification, machine learning, computational biology, computational economics, computational geometry, and computational number theory and algebra“
- Die GTI-Vorlesung behandelt hiervon nur einen kleinen Ausschnitt
  - Die hier behandelten Inhalte sind auch für andere Bereiche der Theoretischen Informatik und für die Informatik insgesamt grundlegend
  - Die Stoffauswahl ist aber ziemlich kanonisch und unterscheidet sich von Uni zu Uni nur wenig

# Inhalt

0.1 Was ist Theoretische Informatik?

▷ **0.2 Themen der GTI-Vorlesung**

0.3 Literatur

0.4 Organisatorisches

# Die 4 Themen der GTI-Vorlesung - A: Reguläre Sprachen

- Die **regulären Sprachen** bilden eine Klasse von einfachen Sprachen mit
  - vielen Anwendungen
  - vielen äquivalenten Charakterisierungen
  - weiteren angenehmen Eigenschaften



- **Beispiele für Anwendungen**

- Bezeichner in Programmiersprachen
- Verifikation zustandsbasierter Systeme
- Schemasprachen für XML
- Zeichenkettensuche (grep)
- Spezifikation zulässiger Eingaben für Web-Formulare

- **Viele Charakterisierungen:**

- Reguläre Ausdrücke
- Endliche Automaten
  - \* deterministisch oder nicht-deterministisch
  - \* 1-Weg oder 2-Weg
  - \* mit oder ohne  $\epsilon$ -Übergänge
- Grammatiken, logische Formeln, ...

- **Angenehme Eigenschaften:**

- einfache Spezifikationssprache:
  -  reguläre Ausdrücke
- einfache Testalgorithmen
  -  endliche Automaten
- Automatische Umwandlung von Spezifikationen in Testalgorithmen
- Optimaler Testalgorithmus konstruierbar
- Methode zum Testen, ob Spezifikationen und Programme äquivalent sind
- Methode zum Nachweis, dass eine Sprache nicht regulär ist

# Die 4 Themen der GTI-Vorlesung - B: Kontextfreie Sprachen

- Die **kontextfreien Sprachen** bilden eine weitere Klasse einfacher Sprachen:
  - ausdrucksstärker als die regulären Sprachen
  - auch viele Charakterisierungen
  - angenehme Eigenschaften

☞ aber nicht mehr ganz so viele

- Hauptanwendung: Syntaktische Struktur von Programmiersprachen

- **Charakterisierungen**
  - Spezifikationssprache: Grammatiken, bzw. Syntaxdiagramme
  - Algorithmen: Kellerautomaten (richtig effizient nur für Teilklassen)

- **Angenehme Eigenschaften**
  - automatische Umwandlung von Spezifikation in Testalgorithmus
  - Methode zum Nachweis, dass eine Sprache nicht kontextfrei ist

# Die 4 Themen der GTI-Vorlesung - C: Berechenbarkeit

- **Hauptfragen:**

- Was heißt „berechenbar“?
- Gibt es algorithmische Probleme, die nicht berechenbar sind?
- Falls ja, wie sehen solche Probleme aus?

- **Was heißt berechenbar?**

- Es gibt viele Modelle, die den Begriff eines Algorithmus und damit Berechenbarkeit formalisieren
- Praktisch alle sind äquivalent!
- Church-Turing-These: Diese Modelle definieren Berechenbarkeit
- Wir werden kennen lernen:
  - \* Turing-Maschinen
  - \* WHILE-Programme
  - \* Rekursive Funktionen

- **Und was ist nicht berechenbar?**

- Wir werden sehen: es gibt viele, auch praktisch relevante Probleme, die sich algorithmisch nicht lösen lassen

# Die 4 Themen der GTI-Vorlesung - D: Komplexitätstheorie

- **Hauptfragen:**

- Was heißt „effizient berechenbar“?
- Gibt es algorithmische Probleme, die berechenbar, aber nicht effizient berechenbar sind?
- Falls ja, wie sehen solche Probleme aus?

- **Was heißt effizient berechenbar?**

- Weitgehender Konsens:
  - \* Rechenzeit wächst höchstens polynomiell in Größe der Eingabe
- Die genannten Berechnungsmodelle sind auch in dieser Hinsicht äquivalent
- Komplexitätsklasse: **P**

- **Wie sehen nicht effizient berechenbare Probleme aus?**

- Es gibt tausende von praktisch relevanten Problemen, für die
  - \* kein effizienter Algorithmus bekannt ist
  - \* kein Beweis, dass sie nicht effizient berechenbar sind, bekannt ist
- Die meisten davon sind im folgenden Sinne äquivalent:
  - \* Hat eines einen effizienten Algorithmus, so haben alle einen
  - \* Hat eines keinen effizienten Algorithmus, so hat keines einen
- **NP**-vollständige Probleme
- **P** = **NP**-Frage



# Sichere Systeme

- Das iPhone-Beispiel illustriert, dass sichere Systeme ein wichtiges Thema für die Informatik darstellen
- Sie werden uns als Querschnittsthema in mehreren Teilen der Vorlesung beschäftigen
- Grundlegendes algorithmisches Problem:
  - Gegeben: ein Modell  $S$  eines Systems und eine formale Beschreibung  $E$  einer Eigenschaft
  - Lässt sich automatisch überprüfen, ob  $S$  die Eigenschaft  $E$  hat?

👉 Model Checking

- Zum Beispiel:
  - $S$  ein Smartphone-OS,  
 $E$ : „Kein Zugang ohne PIN möglich“
  - $S$  ein Programm,  
 $E$ : „ $S$  terminiert für jede Eingabe“
  - $S$  ein Kommunikationsprotokoll,  
 $E$ : „ $S$  führt niemals zu einer Verklemmung“
  - $S$  ein Schaltkreis,  
 $E$  eine Boolesche Funktion, die  $S$  berechnen soll
- Wir werden Fälle sehen, in denen eine solche automatische Überprüfung möglich ist, aber auch andere...

# Was sollen Sie in GTI lernen?

- **Erkenntnisse:**

- Grenzen der Möglichkeiten von Computern: nicht berechenbare Probleme
- Praktische Grenzen von Computern: nicht effizient lösbare Probleme
- ...

- **Fähigkeiten:**

- Abstraktion
- Formalisieren
- Analyse, z.B.: Erkennen von schwierigen Berechnungsproblemen
- ...

- **Grundwissen & Handwerkszeug:**

- Formale Grundlagen des Rechnens mit Computern
- Umgang mit regulären Sprachen, regulären Ausdrücken und Automaten
- Reguläre Sprachen / endliche Automaten „erkennen“
- Theoretische Grundlagen für Übersetzer
- ...

# Das Semester im Überblick

## 0: Einleitung

### A: Reguläre Sprachen

- 1: Reguläre Ausdrücke
- 2: Endliche Automaten
- 3: Äquivalenz der Modelle
- 4: Minimierung endlicher Automaten
- 5: Abschlusseigenschaften, Grenzen und Algorithmen
- 6: Anwendungen und Erweiterungen

### B: Kontextfreie Sprachen

- 7: Kontextfreie Grammatiken
- 8: Normalformen und Korrektheitsbeweise
- 9: Kellerautomaten
- 10: Pumping-Lemma, Algorithmen und Abschlusseigenschaften
- 11: Wortproblem und Syntaxanalyse

- Wrap-Up

### C: Berechenbarkeit

- 12: Verschiedene Berechnungsmodelle
- 13: Die Church-Turing-These
- 14: Unentscheidbare Probleme 1
- 15: Unentscheidbare Probleme 2
- 16: Varianten, Einschränkungen und Erweiterungen

### D: Komplexitätstheorie

- 17: Polynomielle Zeit
- 18: **NP** und **NP**-Vollständigkeit
- 19: **NP**-vollständige Probleme
- 20: **NP**: Weitere Erkenntnisse
- 21: Zufallsbasierte Algorithmen
- 22: Zufallsbasierte Komplexitätsklassen

### Bonustrack

- 23: Parametrisierte Algorithmen und Komplexität
  - Probeklausur

# Inhalt

0.1 Was ist Theoretische Informatik?

0.2 Themen der GTI-Vorlesung

▷ **0.3 Literatur**

0.4 Organisatorisches

# Literatur

- Hopcroft, Motwani, Ullman. Einführung in die Automatentheorie, Formale Sprachen und Berechenbarkeit. Pearson.  
(ältere Auflagen: ... und Komplexitätstheorie)
  - Umfassend, viele Beispiele, gut aufgebaut, sehr verständlich
- Wegener. Theoretische Informatik. Teubner.
  - gut aufgebaut, verständlich, mittlerer Umfang
- Folienskript SoSe 2016, über Moodle
  - Kurz, wenige Beispiele, kostenlos
- G. Vossen, U. Witt: Grundlagen der Theoretischen Informatik mit Anwendungen. Vieweg.
  - Umfassend, viele Beispiele, gut aufgebaut, verständlich
- Schöning. Theoretische Informatik kurz gefasst. Spektrum.
  - gut aufgebaut, sehr verständlich, kürzer als die anderen, nicht so umfassend
- Das Erscheinungsjahr bzw. die Auflage ist jeweils nicht so wichtig (neuer ist natürlich besser...), aber beim erstgenannten Buch sollte unbedingt Herr Motwani einer der Autoren sein

# Inhalt

0.1 Was ist Theoretische Informatik?

0.2 Themen der GTI-Vorlesung

0.3 Literatur

▷ **0.4 Organisatorisches**

# Die Bestandteile der Veranstaltung

- Vorlesung
- Übungsaufgaben
- Übung
- Tutorien
- Helpdesk
- Prüfungen und Studienleistung
- GTI online

# Vorlesung

- **Termine:**

- Dienstag, 10:15 – 11:50 Uhr, HG II, HS 3
- Donnerstag, 12:15 – 13:50 Uhr, HG II, HS 3
- 11.6.-23.6.: im Zelt

☞ jeweils 5 Minuten Pause

- Durchgängiger Einsatz von **Folien**
- Erläuterungen an der Tafel
- Die Folien werden in Moodle bereitgestellt

- **Zweck der Vorlesung:**

- Vermittlung aller wesentlichen Inhalte

- **Gebrauchsanleitung für die Vorlesung:**

- Denken Sie mit
- Stellen Sie Fragen
- Schreiben Sie nur das Nötigste mit
- **Klappen Sie Ihr Notebook bitte zu!**

- **Nachbereitung:**

- Arbeiten Sie die Vorlesung nach
- Geben Sie sich dabei erst zufrieden, wenn Sie jedes Detail jeder Folie (mindestens einmal!) verstanden haben



# Übungsaufgaben (1/3)

- **Zweck der Übungsaufgaben:**

- Inhalte der Vorlesung wiederholen
- Erkennen, wo es mit dem Verständnis noch hapert
- Techniken aus der Vorlesung **anwenden**
- Klausurvorbereitung

- **Anforderungen an die Lösungen**

- Lösungen sind nur vollständig, wenn sie begründet und erklärt werden
- Formale Beweise werden nur erwartet, wenn dies in der Aufgabenstellung ausdrücklich erwähnt wird

- **Gruppenarbeit**

- Sie können die Übungsaufgaben zusammen mit anderen bearbeiten und gemeinsam Lösungswege suchen
- Das **Aufschreiben** der Lösungen erfolgt aber individuell

- **Sanktionen:**

- Wenn mehrere Personen offensichtlich voneinander abgeschriebene Lösungen abgeben, erhalten sie alle 0 Punkte
- Im Wiederholungsfall ist die Studienleistung verwirkt und wir behalten uns weitere Maßnahmen vor

# Übungsaufgaben (2/3)

## • Übungsaufgaben-Zyklus bis zum 6.5.:

👉 bis Blatt 3

- Ausgabe Übungsblatt:
  - \* donnerstags, Woche  $n$
- Abgabe Lösungen:
  - \* donnerstags, Woche  $n+1$   
👉 23:59 Uhr, Öffnungszeiten beachten!
- Besprechung:
  - \* Woche  $n+2$

## • Übungsaufgaben-Zyklus ab dem 7.5.:

👉 ab Blatt 4

- Ausgabe Übungsblatt:
  - \* dienstags, Woche  $n$
- Abgabe Lösungen:
  - \* dienstags, Woche  $n+1$   
👉 23:59 Uhr, Öffnungszeiten!
- Besprechung:
  - \* Woche  $n+1,5$   
👉 Donnerstag-Mittwoch

## • Abgabe und Korrektur:


- Abgabe
  - \* in der Vorlesung
  - \* sonstige Möglichkeiten: siehe Übungsblatt
- Ihre Lösungen werden korrigiert

## • Zu jedem Übungsblatt wird eine **Beispiel-Lösung** veröffentlicht


👉 am Ende von Woche  $n+2$

- Wichtig: es kann mehrere Lösungswege geben, die Beispiel-Lösung repräsentiert meist nur **einen** davon

## Übungsaufgaben (3/3)

- Es gibt 12 Übungsblätter zur Bearbeitung und Abgabe
- Ab Blatt 2 je 15 Punkte je Blatt  in der Regel
- Je Blatt ca. drei Aufgaben, deren Bearbeitung Punkte ergeben kann
- möglicherweise eine schwierigere Zusatzaufgabe, die bei Bearbeitung korrigiert und bepunktet wird, aber nicht besprochen wird

- **Übungsblatt 1**

- Blatt 1 bezieht sich auf den Stoff der Vorlesung vom 12.4. (Donnerstag) und ist etwas weniger umfangreich  9 statt 15 Punkte

# Übung

- In den Übungsstunden sollen erlernt werden:
  - mündliche Präsentation von Inhalten der Theoretischen Informatik
  - die Fähigkeit zur Reflexion von Inhalten der Theoretischen Informatik, auch im Gespräch mit anderen
- Um dieses Ziel zu erreichen
  - werden Präsenzaufgaben bearbeitet
  - werden die Lösungen der Übungsaufgaben von den Studierenden vorgeführt
  - werden alternative Lösungswege und fehlerhafte Ansätze unter den Studierenden diskutiert

- **Termine:** siehe Moodle

- **Beginn:**

- Montag, 16.4.
- In der ersten Übungsstunde wird ein Ü-Blatt mit „Präsenzaufgaben“ besprochen

- **Anmeldung zu den Übungsgruppen:**

- [ess.cs.tu-dortmund.de/ASSESS](http://ess.cs.tu-dortmund.de/ASSESS)
- Anmeldung ab heute, 12:00 Uhr
- bis Freitag, 13.4., 12:00 Uhr
- Bekanntgabe der Gruppeneinteilung am Freitag, 13.4.
- Bitte stellen Sie sicher, dass Ihr Studiengang korrekt ist!

# Tutorien

- **Zusätzliches Angebot:** Tutorium

- **Zweck des Tutoriums:**

- Wiederholung des Stoffes und Prüfungsvorbereitung
- Besondere Unterstützung der „ThIfAI“-Hörer

- **Termine:**

(B2) Mo 14-16, OH12, 1.055

👉 ThIfAI

(A1) Mi 14-16, OH12, E.003

👉 ThIfAI

(A2) Do 10-12, OH14, E 23

(B1) Fr 12-14, OH12, 3.031

- Beginn: Freitag, 13.4. (B1)
- ThIfAI-Hörer haben bei Raumüberfüllung an den gekennzeichneten Terminen Vorrang

- **Themen:**

- Werden jeweils in Moodle bekannt gegeben
  - \* In den beiden A-Tutorien und den beiden B-Tutorien jeweils das gleiche Thema
- Themenvorschläge sind willkommen (INPUD)

## Helpdesk

- Lerngruppe finden
- Lösungswege für Aufgaben diskutieren
- Studentische Hilfskräfte sind zu gewissen Zeiten anwesend, um
  - einfache Fragen zu beantworten
  - Diskussionen zu unterstützen
- ... aber **nicht** um
  - Übungsaufgaben zu lösen
  - Lösungen zu überprüfen
- SHKs geben ihre Hinweise nach bestem Wissen
- Eine Garantie für Korrektheit wird nicht gegeben
- SHK-Zeiten werden im Moodle bekannt gegeben
- Auf jeden Fall: montags, dienstags und donnerstags
- Am Anfang des Semesters: mittwochs
- Später: freitags
- Beginn: 16.4.

# Prüfung und Studienleistung (1/2)

- Klausurtermine:
  - 1. Klausur: 13.8.2018, 8:00-11:00 Uhr
  - 2. Klausur: 27.9.2018, 12:00-15:00 Uhr
- **Erlaubte Hilfsmittel:** zwei DIN A4 Blätter mit selbst erstellten handschriftlichen Notizen
  - Alle Seiten haben einen oberen Rand von 2 cm  
☞  $\geq$  ein 10-Cent-Stück
  - Auf allen vier Rändern Name und Matrikelnummer eintragen
- Anmeldung zur Klausur im BOSS
  - **Vergewissern Sie sich bitte, dass die Anmeldung im BOSS geklappt hat!!!**
  - **Vergewissern Sie sich bitte, dass die Anmeldung im BOSS geklappt hat!!!**
  - **Vergewissern Sie sich bitte, dass die Anmeldung im BOSS geklappt hat!!!**
  - Studierende, für deren Studiengänge keine Anmeldung im BOSS möglich ist, melden sich direkt bei uns an ☞ nähere Informationen folgen

## Prüfung und Studienleistung (2/2)

- Voraussetzung zur Teilnahme an der Klausur ist das Erbringen der **Studienleistung**:
  - Erreichen von
    - \*  $\geq 28$  von 84 erreichbaren Punkten der Aufgabenblätter 1-6
    - \*  $\geq 30$  von 90 erreichbaren Punkten der Aufgabenblätter 7-12
- Mein Tipp: Verlassen Sie sich nicht auf das jeweils letzte Übungsblatt
- Die Punkte sind jeweils gut mit den ersten vier Blättern zu erreichen
- Die Aufgaben werden später nicht leichter
- Zum Erwerb der genannten Kompetenzen und als Vorbereitung zur Klausur empfehlen wir dringend die aktive Teilnahme an den Übungsgruppen

- Die Studienleistung muss nicht erbracht werden von:
  - Studierenden des Diplomstudiengangs
  - Studierenden, die die Studienleistung im letzten SoSe erbracht haben
  - Studierenden, die die Studienleistung irgendwann erbracht haben **und** bereits eine Klausur geschrieben haben
- Dringende Empfehlung für diese Gruppen: nehmen Sie aktiv am Übungsbetrieb teil



# GTI online

- Auf der **Vorlesungsseite** finden sich:
  - die Vorlesungsankündigung
  - Allgemeine Informationen zu den Klausuren

- **Moodle:**
  - moodle.tu-dortmund.de
  - Dort finden Sie
    - \* Vorlesungsfolien und evtl. weitere Materialien
    - \* Übungsblätter
    - \* Beispiellösungen
    - \* Ankündigungen
  - Bitte anmelden!

- Die Vorlesungsfolien gibt es in drei Varianten:
  - Zum Anschauen am Bildschirm
    - ☞ jeder „Klick“ auf Extra-Seite
  - Zum Ausdrucken
  - Zum Ausdrucken mit reduziertem Farbeinsatz
  - Öko-Tipp: 4-auf-1 ist noch gut lesbar

- Hinweis: nicht alle Abbildungen/Fotos werden in den Foliensätzen zur Verfügung gestellt

- Im INPUD-Forum gibt es eine moderierte Diskussionsgruppe zu GTI
- Dort können Sie Fragen stellen, mit anderen über Vorlesung und Übung diskutieren, sich Lernpartner suchen usw.

Zu guter Letzt...

Viel Erfolg!

Und auch etwas Spaß!