

# Modul Modellierung WS 2024/25 – Lernziele

(nach Modulbeschreibung)

- ▶ Anwendung mathematischer und logischer Grundkonzepte zur Modellierung praktischer Aufgabenstellungen
- ▶ Beherrschen von Formalismen zur Beschreibung von Anforderungen an Software und Systeme (Spezifikation)
- ▶ Wissen um Möglichkeit formaler Korrektheitsnachweise für Software und Systeme (Verifikation)

Nebenwirkungen:

- ▶ Fähigkeit zur Abstraktion  
(durch konsequente Verwendung formaler Darstellungen)
- ▶ anwendungsbereite Kenntnisse zur Modellbildung  
(durch viele Beispiele und Übungsaufgaben)
- ▶ Verweise auf Zusammenhänge zu anderen Gebieten der Informatik und zur Mathematik

# Prozess beim Lösen von Aufgaben (Problemen)

**Analyse** der (informalen) Aufgabe, Identifikation von

- ▶ Aufgabenbereich (Kontext)
- ▶ Eingabedaten (Typ, mögliche Werte)
- ▶ gewünschte Lösung  
(Typ, Eigenschaften, Zusammenhang mit Eingabe)

**Modellierung** (Spezifikation, formale Darstellung) von

- ▶ Aufgabenbereich (Kontext)
- ▶ Anforderungen an Eingaben
- ▶ Anforderungen an Lösungen

Modellierung von (Anforderungen an)

**Daten, Zusammenhängen und deren Eigenschaften**

**Entwurf einer Lösungsstrategie** für die modellierte Aufgabe

(mit vorhandenen oder neuen Methoden)

Modellierung von (Anforderungen an)

**Abläufen und deren Eigenschaften**

**Umsetzung** der Lösungsstrategie im Modellbereich

**Ausführung** der Lösungsstrategie im Modellbereich

**Übertragung** der Lösung vom Modellbereich in die Realität

## Beweise in der Informatik

Dijkstra sagt dazu in

<http://www.cs.utexas.edu/~EWD/transcriptions/EWD13xx/EWD1305.html>:

A programmer has to be able to demonstrate that his program has the required properties. If this comes as an afterthought, it is all but certain that he won't be able to meet this obligation: only if he allows this obligation to influence his design, there is hope he can meet it. Pure a posteriori verification denies you that whole-some influence and is therefore putting the cart before the horse, but that is exactly what happens in the software houses where "programming" and "quality assurance" are done by different groups.

[Needless to say, those houses deliver without warranty.]

aus dem Edsger W. Dijkstra Archiv

<http://www.cs.utexas.edu/~EWD/>

# Modul Modellierung WS 2024/25 – Inhalt

Modellierung von Daten und Zusammenhängen durch

- ▶ Mengen (Individuen-, Wertebereiche), Multimengen
- ▶ Relationen
- ▶ Graphen
- ▶ Funktionen
- ▶ Terme, strukturelle Induktion
- ▶ algebraische Strukturen
- ▶ Abstrakte und konkrete Datentypen

Modellierung von Eigenschaften durch Logiken

- ▶ klassische Aussagenlogik
- ▶ klassische Prädikatenlogik der ersten Stufe

Modellierung von Abläufen durch

- ▶ Algorithmen
- ▶ Zustandsübergangssysteme (sequentiell)

# Mengen

- ▶ extensionale, intensionale Darstellung
- ▶ (endliche und unendliche) Mächtigkeit  $|M|$
- ▶ Russells Paradox
- ▶ Leere Menge  $\emptyset$
- ▶ Mengenoperationen:  $\cup, \cap, \setminus, \bar{\phantom{x}}$
- ▶  $\times, ^n$ , Tupel, Vektoren
- ▶ Mengenbeziehungen:  $\in, \subseteq$
- ▶ zusammengesetzte Wertebereiche
- ▶  $B^A = \{f : A \rightarrow B\}$ ,
- ▶ Potenzmenge  $2^A$
- ▶ charakteristische Funktion  $\chi_A$
- ▶ Multimengen

# Wörter und Sprachen

- ▶ Alphabet  $A$
- ▶  $A^*$ , Wörter (Folgen),  $\varepsilon$
- ▶ intensionale, extensionale Darstellung
- ▶ Länge  $|w|$ , Anzahl von Buchstabenvorkommen  $|w|_a$
- ▶ Operationen  $\circ$ ,  $R$  auf Wörtern
- ▶ Präfix-, Postfix-, Infix-Beziehungen
- ▶ (quasi-)lexikographische Ordnung

## Sprachen

- ▶ Sprachen als Mengen
- ▶ Operationen  $\circ$ ,  $*$ ,  $R$  auf Sprachen
- ▶ reguläre Ausdrücke
- ▶ reguläre Sprachen
- ▶ Zusammenhang mit endlichen Automaten

# Relationen

- ▶ Identität auf der Mengen  $M$ :  $I_M = \{(x, x) \mid x \in M\}$
- ▶ Operationen:
  - ▶ Verkettung
  - ▶ inverse Relation
  - ▶ Projektionen
  - ▶ Einschränkung
- ▶ Darstellung als Funktion
- ▶ Eigenschaften (reflexiv, transitiv, ...)
- ▶ Hüllen
- ▶ Quasiordnungen
- ▶ Halbordnungen, Hasse-Diagramm, minimale / maximale Elemente, Minimum / Maximum
- ▶ Äquivalenzrelationen
- ▶ Zerlegungen von Mengen in Äquivalenzklassen

# Graphen

- ▶ gerichtete / ungerichtete Graphen
- ▶ Darstellungsformen
- ▶ Eigenschaften
- ▶ Isomorphie
- ▶ spezielle Graphenklassen  $I_n, P_n, C_n, K_n, K_{m,n}$
- ▶ (induzierte, aufspannende) Teilgraphen
- ▶ Graph-Operationen
- ▶ Wege, Pfade, Kreise in Graphen
- ▶ Nachbarschaften, Zusammenhang, Zusammenhangskomponenten
- ▶ Euler-Weg und -Kreis,
- ▶ Hamilton-Pfad und -Kreis
- ▶ bipartite Graphen, Färbungen (der Knoten und Kanten)
- ▶ chromatische Zahl, Cliquenzahl

# Funktionen

- ▶ als spezielle Relationen
- ▶ Definitions- und Wertebereich
- ▶ totale / partielle Funktionen
- ▶ Bild
- ▶ Urbild
- ▶ Inverse
- ▶ Verkettung von Funktionen
- ▶ Eigenschaften (assoziativ, kommutativ)
- ▶ Folgen, Mengen, Multimengen als Funktionen
- ▶ Darstellung von Relationen als Funktionen

# Terme

- ▶ Sorten (Typen)
- ▶ Signaturen (einsortig und mehrsortig)
- ▶ Terme, Grundterme (einsortig und mehrsortig)
- ▶ Baumdarstellung
- ▶ strukturelle Induktion
- ▶ Teilterm-Relation

# (Algebraische) Strukturen

- ▶  $\Sigma$ -Struktur zu Signatur  $\Sigma$
- ▶ einsortig / mehrsortige Strukturen
- ▶ Trägermengen (Universum, Individuenbereich)
- ▶ Interpretation der Signatur-Symbole
- ▶ Werte von Termen und Formeln in Strukturen
- ▶ Äquivalenz von Termen und Formeln in Strukturen
- ▶ konkrete Datentypen

# Logiken zur Modellierung verschiedener Aufgabenbereiche

verschiedene Logiken unterschieden sich in

- ▶ Ziele:
  - ▶ Repräsentation von Informationen (z.B. Wissensrepräsentation)
  - ▶ Formalisierung von Anforderungen (z.B. Entwicklung von Systemen, Hard-, Software)
- ▶ Ausdrucksstärke:
  - ▶ Aussagen über welche Bereiche lassen sich überhaupt formulieren?
  - ▶ kompakte Darstellung typischer Aussagen
  - ▶ Verständlichkeit der Formalisierung
- ▶ algorithmische Entscheidbarkeit wichtiger Fragen (maschinelle) Verarbeitung von Informationen

# Klassische Aussagenlogik

## Syntax:

- ▶ Junktoren
- ▶ aussagenlogische Formeln (Baumstruktur)
- ▶ strukturelle Induktion
- ▶ Teilformeln
- ▶ syntaktische Umformungen
- ▶ syntaktisches Schließen (Resolutionskalkül)

## Semantik:

- ▶ Wahrheitswertfunktionen
- ▶ Belegungen von Aussagenvariablen
- ▶ Wahrheitswerttabellen
- ▶ Modelle, Modellmenge von Formeln und Formelmengen
- ▶ erfüllbar, unerfüllbar, allgemeingültig
- ▶ semantische Äquivalenz
- ▶ semantische Folgerungen
- ▶ syntaktisches Schließen,  
aussagenlogischer Resolutionskalkül
- ▶ beschränkte Ausdruckstärke

# Klassische Prädikatenlogik

## Syntax:

- ▶ Quantoren
- ▶ freie, gebundene Variablen
- ▶ Struktur der Atome
- ▶ Formeln (Baumstruktur, Schichten)
- ▶ syntaktische Umformungen
- ▶ Aussagenlogik als Spezialfall

## Semantik:

- ▶ Belegungen von Individuenvariablen
- ▶ Interpretationen (Struktur, Belegung)
- ▶ Modelle, Modellmenge von Formeln und Formelmengen
- ▶ (un-)erfüllbare, allgemeingültige Formeln
- ▶ Charakterisierung von Strukturen durch Satzmengen
- ▶ semantische Äquivalenz
- ▶ semantische Folgerungen
- ▶ Aussagenlogik als Spezialfall

# Vergleich verschiedener Logiken

klassische Logiken:

- ▶ klassische Prädikatenlogik der ersten Stufe FOL:
  - ▶ hohe Ausdrucksstärke,
  - ▶ kompakte, verständliche Formulierungen,
  - ▶ unentscheidbar
- ▶ klassische Aussagenlogik AL (Spezialfall, Fragment von FOL):
  - ▶ geringe Ausdrucksstärke
  - ▶ umfangreiche Darstellung
  - ▶ entscheidbar
  - ▶ SAT-Solver zur Aufgabenlösung

Ausblick:

Speziallogiken für verschiedene Anwendungsbereiche, z.B.

- ▶ Temporallogiken (Spezifikation, Verifikation von Systemen)
- ▶ Hoare-Logik (Spezifikation, Verifikation imperativer Software)
- ▶ Beschreibungslogiken (Wissensextraktion aus Ontologien, Semantic Web)

oft entscheidbare Fragmente von FOL mit modifizierter Syntax  
mehr dazu in Lehrveranstaltungen später im Studium

# Abstrakte Datentypen (ADT)

Definition der

- ▶ Syntax:
  - ▶ Sorten
  - ▶ Signatur
- ▶ Semantik: Axiome (oft Menge prädikatenlogischer Sätze)

ADT (Software-Schnittstellen)

als Repräsentation von

Mengen konkreter Datentypen (Implementierungen)

Nachweis zusätzlicher Eigenschaften durch Folgerungen aus der Axiomenmenge

Beispiele:

- ▶ ADT Himmelsrichtungen, ADT Menge, ADT Stack
- ▶ verschiedene passende konkrete Datentypen

# Algorithmen

- ▶ Grundbausteine
- ▶ Verknüpfungen (Ablauf-Strukturen)
- ▶ Ausführung durch Aktoren (z.B. Maschine)
- ▶ Ausführung als Zustandsübergangssystem
- ▶ Nebenwirkungen
- ▶ Formale Spezifikation (Vor- und Nachbedingungen):  
Anforderungen an die Wirkungen von Algorithmen

# Modellierung von Abläufen

## Zustandsübergangssysteme:

- ▶ Zustände
- ▶ Zustandsübergangs-Relation
- ▶ Darstellung als Graph mit Kantenfärbung
- ▶ formale Beschreibung von Zuständen und Übergängen
- ▶ Formalisierung von Anforderungen an Systeme  
Spezifikation und Verifikation von Systemen (Ausblick)
- ▶ Automaten mit Zusatzspeicher (Ausblick)
- ▶ Ausführung imperativer Programme als  
Zustandsübergangssystem  
Verifikation imperativer Programme (Ausblick)

# Modul Modellierung WS 2024/25 – Organisation

vorgesehener Zeitaufwand laut Modulbeschreibung:

**Präsenzstudium:** 90 h als 4+2 SWS = 6 h jede Woche

**Selbststudium:** 156 h

10 h / Woche: Nach- und Vorbereitung,  
Übungsaufgaben

16 h Vorbereitung Prüfung

Bedingungen für Prüfungszulassung:

**Opal:**  $\geq \lfloor 62 \cdot 70\% \rfloor = 43$  Punkte und

**Seminar-Vorträge:**  $\geq 3$  Punkte und

**Autotool:**  $\geq \lfloor 36/2 \rfloor = 18$  Punkte

118 erfüllte PVL

Individuelle Zulassungen zur Prüfung sind in Opal zu sehen.

# Prüfung

**Wann?** Dienstag, 25. 2. 2025 um 13:30 - 15:30 Uhr

**Wo?** in mehreren Hörsälen :  
NI001-H, TRA 140-H, FE006-H  
Wegweiser zur Raumverteilung:  
im Foyer FE (ab ca. 13:00 Uhr)

**Wie?** Klausur 120 min  
(einzige) zugelassene Hilfsmittel:  
ein A4-Blatt beidseitig handbeschrieben

**Was?** Inhalt der LV  
Aufgabentypen aus Übungsserien bekannt

**Viel Erfolg!**