



Zentralübung zu
Einführung in die Informatik I

Dr. Christian Herzog
Technische Universität München

Wintersemester 2000/2001
20. November 2000

Übersicht

- ❖ Organisatorisches
- ❖ Einführung in die Programmierung mit Java
 - Anforderungen/Vorwissen/Vorgehensweise
 - Konstruktoren und Operationen
- ❖ Vorbereitung von Übungsblatt 4
 - Wiederholung Algorithmus
 - Wiederholung Zeichensequenzen
 - Wiederholung Textersetzungs-system
 - Wiederholung Markov-Algorithmus

Organisatorisches

- ❖ Bitte halten Sie sich auf dem Laufenden:
 - regelmäßiges (am besten tägliches) „Vorbeischauen“ auf der Homepage zu Vorlesung und Übung im WWW, um Mitteilungen der Übungsleitung zu erhalten
 - regelmäßiges (am besten tägliches) Lesen Ihrer email, z.B. um Nachrichten Ihrer Tutorin bzw. Ihres Tutors zu erhalten.

Java und „Einführung in die Informatik I“

- ❖ Java ist die Programmiersprache, die in Vorlesung und Übung verwendet wird.
- ❖ Sie wird dort auch schrittweise eingeführt.
- ❖ Java ist für Programmier-Anfänger nicht ganz leicht:
 - Auch das einfachste Java-Programm enthält Konstrukte, die dem Neuling nicht leicht verständlich sind.
 - Das ist dem Dozenten, der Übungsleitung, den Tutorinnen und Tutoren bekannt und wird von ihnen berücksichtigt.
 - Deshalb: Vertrauen Sie sich unserer Führung an!
 - Wir erwarten von Ihnen keine Vorkenntnisse in Java.
 - Wir erwarten von Ihnen (vorerst) auch nicht, dass Sie sich Java aus Büchern selbst beibringen.
 - Ein Tutorium wie „Java in 21 Tagen“ ist zurzeit für Anfänger wenig hilfreich.

Unsere Vorgehensweise

❖ Generell:

- Sie programmieren das, was Sie können bzw. lernen sollen. Der Rest wird zur Verfügung gestellt.
- Die zur Verfügung gestellten Programme bzw. Programmkonstrukte müssen Sie nicht verstehen.

❖ Aktuelle Phase:

Einige erste Sprachkonstrukte, um die ersten Klassen erstellen zu können (einfache Methodendeklaration, Methodenaufruf, Konstruktoraufruf, extends).

❖ Ende November/Anfang Dezember:

Sprachkonstrukte, die zum Programmierparadigma „funktionales Programmieren“ gehören.

❖ Dezember:

Sprachkonstrukte, die zum Programmierparadigma „imperatives Programmieren“ gehören.

Unsere Vorgehensweise (Fortsetzung)

- ❖ Anfang 2001:
Sprachkonstrukte, die zum Programmierparadigma „objektorientiertes Programmieren“ gehören.
- ❖ Danach:
 - von uns: einige weitere Sprachkonstrukte
 - von Ihnen: Vertiefung der Kenntnisse durch Selbststudium
- ❖ Das Wichtigste dabei:
 - Üben, üben, üben (learning by doing)!

Konstruktoren

- ❖ Konstruktoren erzeugen Instanzen einer Klasse
 - Erinnerung - Definition Instanz:
Ein Objekt ist eine Instanz einer Klasse K , wenn es Element der Menge aller Objekte der Klasse K ist.
- ❖ Konstruktoren bei (abstrakten) Rechenstrukturen:
 - createStack: \rightarrow Keller(T)
 - createStack: \rightarrow Keller(-)
 - create: \rightarrow Sequenz(char)
 - erzeugeStudent: $\text{String} \times \text{String} \times \text{double} \rightarrow$ Student
 - erzeugeStudentenSequenz: \rightarrow StudentenSequenz
- ❖ Konstruktoren erzeugen Instanzen und initialisieren deren Attribute mit
 - voreingestellten Werten (Default-Werten) oder
 - Werten, die als Parameter übergeben werden.

Konstruktor in Java (Wiederholung)

- ❖ Operation einer Klasse, die
 - keinen Ergebnistyp hat;
 - den Namen der Klasse trägt.
- ❖ Beispiel (siehe Programmieraufgabe 9 von Übungsblatt 3):

```
– class StudentenSequenz extends Object {  
    ... // hier ausgelassen: Deklaration der Attribute  
  
    // Parameterloser Standard-Konstruktor:  
    StudentenSequenz( ) {...}  
  
    ... // hier ausgelassen: Deklaration weiterer Operationen  
  
}
```

Konstrukturen in Java (Fortsetzung)

❖ Weiteres Beispiel aus Aufgabe 9:

```
– class Student extends Object {
```

```
    ...           // hier ausgelassen: Deklaration der Attribute
```

```
    // Parametrisierter Konstruktor:
```

```
    Student (String nachname, String vorname, double note) {...}
```

```
    ...           // hier ausgelassen: Deklaration weiterer Operationen
```

```
}
```

Erzeugung einer Instanz in Java

- ❖ In Java wird die Instanz einer Klasse K erzeugt durch
 - Aufruf eines Konstruktors von K
 - in Verbindung mit dem Schlüsselwort `new`
- ❖ Beispiele
 - `new StudentenSequenz()`
liefert eine neue Instanz der Klasse `StudentenSequenz`
 - `new Student("Zufall", "Rainer", 1.0)`
liefert eine neue Instanz der Klasse `Student` mit entsprechend initialisierten Attributen
 - `nimmAuf (new StudentenSequenz (),
 new Student("Zufall", "Rainer", 1.0))`
 - `StudentenSequenz erzeugeStudentenSequenz () {
 return new StudentenSequenz ();
}`

Was sollten Sie wissen?

❖ zu Konstruktoren in Java:

- Aufbau der Kopfzeile
- Rumpf nur, falls er leer ist
- Aufruf mit new und aktuellen Parametern

❖ zu Operationen (Methoden) in Java:

- Aufbau der Kopfzeile
- Rumpf nur, falls er aus einer einzigen return-Anweisung besteht
- Aufruf mit aktuellen Parametern

Vorbereitung von Übungsblatt 4

- ❖ Wiederholung Algorithmus
- ❖ Wiederholung Zeichensequenzen
- ❖ Wiederholung Textersetzungssystem
- ❖ Wiederholung Markov-Algorithmus

Algorithmus: Definition aus der Vorlesung

❖ Definition Algorithmus:

– Ein Algorithmus ist ein Verfahren zur Verarbeitung von Daten mit einer präzisen, endlichen Beschreibung unter Verwendung effektiver Arbeitsschritte

❖ Präzise: In einer eindeutigen Sprache abgefasst

❖ Endlich: In einer endlichen Form beschrieben

❖ ~~Effektiv: Die Arbeitsschritte sind tatsächlich ausführbar~~

❖ Ein Algorithmus muss nicht terminieren
(im Gegensatz zu Definitionen anderer Autoren)

Taxonomie von Algorithmen (aus Vorlesung)

Algorithmen unterscheiden sich nach

- ❖ der Anwendbarkeit von Regeln
 - Terminierender Algorithmus
 - Nichtterminierender Algorithmus
 - Deterministischer Algorithmus
 - Indeterministischer Algorithmus
- ❖ der Beziehung zwischen Ein- und Ausgabe
 - Determinierter Algorithmus
 - Indeterminierter Algorithmus

Zeichensequenzen: Definitionen aus der Vorlesung

❖ Wort:

Für eine gegebene Menge V von Zeichen bildet eine Folge $x = x_1, \dots, x_n$ mit $x_i \in V$ ein Wort der Länge n . Wir schreiben auch $x = x_1 \dots x_n$ für das Wort und sprechen von einem n -Tupel.

❖ V^n bezeichnet die Menge aller n -Tupel

– $V^n =_{\text{def}} V \times V \dots \times V$ (n -faches Kreuzprodukt über V)

❖ ε bezeichnet die leere Sequenz („leeres Wort“)

❖ V^+ bezeichnet die Menge aller Worte über V ohne das leere Wort.

❖ V^* bezeichnet die Menge aller endlichen Zeichensequenzen von Zeichen aus V .

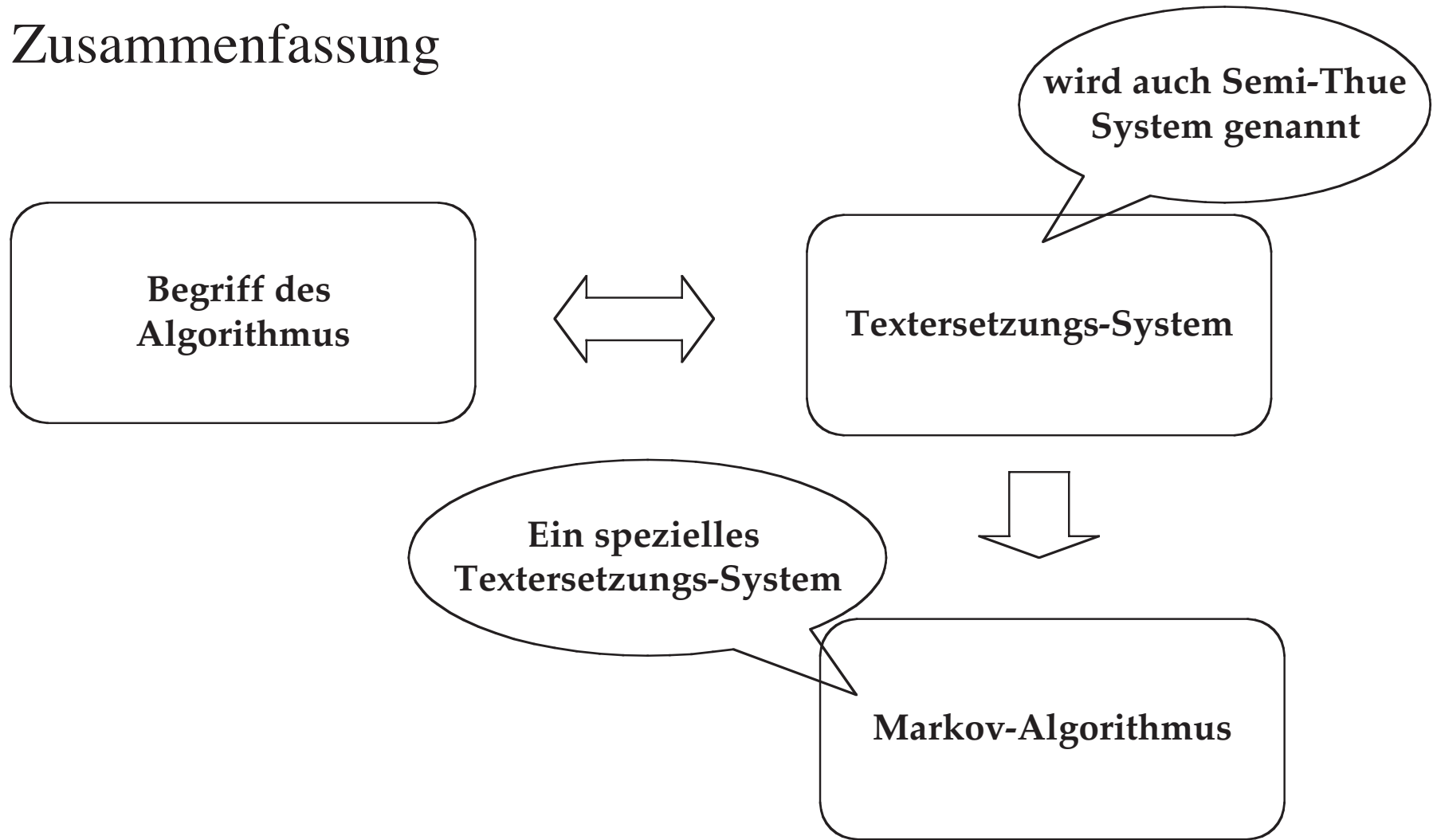
Textersetzungssystem: Definition aus der Vorlesung

- ❖ Eine Menge $T = \{a \rightarrow b, c \rightarrow d, \dots\}$ von Regeln über einem Zeichenvorrat V heißt Textersetzungssystem (auch Semi-Thue System) wenn die folgenden Metaregeln gelten:
 - Sei x ein Wort über V mit dem Teilwort $a \circ \dots \circ b$, d.h.
$$x = x_1 \circ x_2 \circ a \circ \dots \circ b \circ x_n$$
 - Wenn $a \circ \dots \circ b \rightarrow c \circ \dots \circ d$ eine Regel von T ist, dann kann man das Teilwort $a \circ \dots \circ b$ in x durch $c \circ \dots \circ d$ ersetzen.
 - $x_1 \circ x_2 \circ a \circ \dots \circ b \circ x_n \Rightarrow x_1 \circ x_2 \circ c \circ \dots \circ d \circ x_n$
 - Wenn $a \circ \dots \circ b$ mehrfach vorkommt oder mehrere Regeln anwendbar sind, so kann man das Teilwort bzw. die Regel beliebig wählen.
 - Die Regeln können beliebig oft angewandt werden.
 - \Rightarrow heißt direkte Ableitung
- ❖ Textersetzungssystem: einfache Form eines Algorithmus

Markov Algorithmen: Definition aus Vorlesung

- ❖ Definition: Ein Markov-Algorithmus ist ein deterministisches Textersetzungssystem mit endlich vielen Regeln, sowie einigen speziellen Regeln, Zeichen und 2 Metaregeln:
 - Markov-Algorithmen enthalten spezielle Regeln, auch haltende Regeln $x \rightarrow y$ genannt, die durch einen Punkt nach dem Pfeil \rightarrow gekennzeichnet sind.
 - Markov-Algorithmen enthalten zusätzliche Zeichen a, b, c, \dots , sogenannte Schiffchen.
- ❖ Markov-Algorithmen haben immer 2 spezielle Anweisungen ("Metaregeln") für die Ausführung von Regeln:
 - 1. Wähle in jedem Schritt die erste anwendbare Regel. Falls sie auf mehrere Teilwörter anwendbar ist, wende sie auf das am weitesten links stehende Teilwort an.
 - 2. Wende die Regeln solange an, bis eine haltende Regel angewandt wurde, oder bis keine Regel mehr anwendbar ist.

Zusammenfassung



**Vielen Dank für die
Aufmerksamkeit!**