

# Informatik I: Einführung in die Programmierung

## 30. Ausblick

Albert-Ludwigs-Universität Freiburg



Bernhard Nebel

13.02.2016

# 1 Was haben wir gelernt?



Was haben wir gelernt?

Algorithmusbegriff

Was geht nicht?

13.02.2016

B. Nebel – Info I

3 / 17

# Was haben wir gelernt?



Was haben wir gelernt?

Algorithmusbegriff

Was geht nicht?

- Programmieren – jedenfalls ein bisschen
- Python-Programme lesen und verstehen
- Die Informatikperspektive auf die Welt einnehmen können
- Spaß und Begeisterung am Verstehen von (formalen und realen) Problemen
- ... und am Finden von Lösungen (= Programme schreiben)
- Ein paar Hintergründe haben wir auch kennen gelernt,
- z.B. zur Bedeutung einer Zivilklausel – **Do not mention the war!**  
<http://www.youtube.com/watch?v=yf16Lu3xQW0>  
*Fawlty Towers*

13.02.2016

B. Nebel – Info I

4 / 17

# Was wird die Klausur von Ihnen verlangen?



Was haben wir gelernt?

Algorithmusbegriff

Was geht nicht?

- Es wird nicht schwieriger werden als in der Weihnachtsklausur – aber natürlich kommt der Stoff nach Weihnachten dazu
- Schauen Sie sich die Übungsaufgaben noch einmal an
- Auch die Folien/Aufzeichnungen sollten Sie noch einmal konsultieren
- Bei Unklarheiten: Schauen Sie in Python-Bücher und/oder stellen Sie die Fragen in den beiden angebotenen Fragestunden
- Apropos: Algorithmusbegriff

13.02.2016

B. Nebel – Info I

5 / 17

## 2 Algorithmusbegriff



Was haben wir gelernt?  
Algorithmusbegriff  
Was geht nicht?

## Der Algorithmusbegriff in der Informatik-Literatur (1)



Was haben wir gelernt?  
Algorithmusbegriff  
Was geht nicht?

Christos Papadimitriou in *Computational Complexity*, 1994:

*An algorithm is a detailed step-by-step method for solving a problem. But what is a problem? We introduce in this chapter three important examples.*

...

Robert Sedgewick in *Algorithms*, 2011, 4th ed.:

*The term algorithm is used in computer science to describe a finite, deterministic, and effective problem-solving method suitable for implementation as a computer program.*

## Der Algorithmusbegriff in der Informatik-Literatur (2)



Was haben wir gelernt?  
Algorithmusbegriff  
Was geht nicht?

Thomas Ottmann und Peter Widmayer in *Algorithmen und Datenstrukturen*, 2011, 5. Aufl.:

**Was ist ein Algorithmus?** Dies ist eine philosophische Frage, auf die wir in diesem Buch keine präzise Antwort geben werden. Dies ist glücklicherweise auch nicht nötig. Wir werden nämlich in diesem Buch (nahezu) ausschließlich positive Aussagen über die Existenz von Algorithmen durch explizite Angabe solcher Algorithmen machen. Dazu genügt ein intuitives Verständnis des Algorithmusbegriffs und die Einsicht, dass sich konkret angegebene Algorithmen etwa in einer höheren Programmiersprache wie Pascal formulieren lassen. Erst wenn man eine Aussage der Art „Es gibt **keinen** Algorithmus, der dieses Problem löst“ beweisen will, benötigt man eine präzise formale Fassung des Algorithmusbegriffs.

## Der Algorithmusbegriff und die Turing-Maschine



Was haben wir gelernt?  
Algorithmusbegriff  
Was geht nicht?

- Wir hatten in der letzten Vorlesung die Church-Turing-These kennen gelernt:

*Die Klasse der Turing-berechenbaren Funktionen stimmt mit der Klasse der intuitiv berechenbaren Funktionen überein.*

- Dies kann man verstehen als: Algorithmen (zur Berechnung von Funktionen) im intuitiven Sinne entsprechen im formalen Sinne Turing-Maschinen!
- Aber was sind Turing-Maschinen?
- Turing-Maschinen bestehen aus einem **Ein-/Ausgabe-Band**, einem **Lese-/Schreib-Kopf** und einer **Zustandsmaschine**
- Beispiel:  
<http://www.youtube.com/watch?v=gJQTFhkhwPA>

## Algorithmen im engeren Sinne



- Algorithmen entsprechen also Turingmaschinen, wobei man zeigen kann, dass höhere Programmiersprachen und Turingmaschinen i.w. äquivalent sind.
- Wir erinnern uns; ein Algorithmus soll die folgenden Eigenschaften besitzen:

<b>Präzision</b>	Die Bedeutung jedes Einzelschritts ist eindeutig festgelegt.
<b>Effektivität</b>	Jeder Einzelschritt ist ausführbar.
<b>Finitheit (statisch)</b>	Die Vorschrift ist ein endlicher Text.
<b>Finitheit (dynamisch)</b>	Bei der Ausführung wird nur endlich viel Speicher benötigt.
<b>Terminierung</b>	Die Berechnung endet nach endlich vielen Einzelschritten – für alle legalen Eingaben.

- Bei einer Turingmaschine immer erfüllt!

Was haben wir gelernt?

Algorithmusbegriff

Was geht nicht?

## Algorithmen im weiteren Sinne



- Man kann und hat viele Bedingungen gelockert, ohne den Bereich der Turing-Berechenbarkeit zu verlassen:
  - Ein Algorithmus muss nicht immer einen festgelegten Nachfolgezustand / nächsten Schritt haben, sondern die Entscheidung über den nächsten Schritt kann gewürfelt werden (**randomisierter** Algorithmus).
  - In einem Algorithmus können mehrere nächste Schritte möglich sein, wobei der ausgewählt wird, der zum Erfolg führt (**nicht-deterministischer** Algorithmus).
  - Bedingt durch randomisierte oder nicht-deterministische Entscheidungen kann ein Algorithmus verschiedene Werte bei gleichen Eingaben ausgeben (im Falle von **Suchproblemen**, z.B. CSP, oder auch „fehlerhafte Ergebnisse“ geben – bei **Monte-Carlo-Algorithmen**).
- Aber in allen Fällen können wir auch das auf normale TMs zurückführen, d.h. **Determinismus** und **Determiniertheit** sind nicht wirklich notwendig.

Was haben wir gelernt?

Algorithmusbegriff

Was geht nicht?

## 3 Was geht nicht?



Was haben wir gelernt?

Algorithmusbegriff

Was geht nicht?

## Nicht-Berechenbarkeit und Unentscheidbarkeit



- Ein **formaler Algorithmusbegriff** ist vor allem deswegen wichtig, weil wir ja auch **Unmöglichkeitsergebnisse** beweisen wollen.
- Was können wir also z.B. nicht berechnen/entscheiden?
- Gegeben ein beliebiges *Python-Programm*  $\Pi$ , gibt das Programm  $\Pi$  bei jeder Eingabe nach endlicher Zeit ein Ergebnis aus?
- Dies Problem (wie auch jedes ähnliche Problem) ist **unentscheidbar!**
- D.h. wir können **kein** Programm schreiben, um dieses Problem zu entscheiden („ja“ oder „nein“ ausgeben)
- Dies sind alles Dinge, die erst in Info III behandelt werden

...

Was haben wir gelernt?

Algorithmusbegriff

Was geht nicht?

- Programmieren ist zwar „nur“ Handwerkszeug.
- Aber ohne Handwerkszeug geht es in der Informatik nicht
- ... weder in der Praxis noch in der Wissenschaft.
- Wir haben in dieser Vorlesung 3 verschiedene Programmierparadigmen gelernt
- ... schauen Sie sich nochmal an, was ihre Charakteristika sind!

Was haben wir gelernt?

Algorithmus-begriff

Was geht nicht?

- Python ist eine wirklich tolle Programmiersprache!
- Monthy Python ist eine Comedy-Truppe, die (fast?) nicht zu überbieten ist.
- ... und es gibt die Liste der 10 besten Sketches, wobei ich speziell auch bei dem ersten Platz zustimme:  
<https://www.youtube.com/watch?v=2ChPAqPdDdw>

Was haben wir gelernt?

Algorithmus-begriff

Was geht nicht?