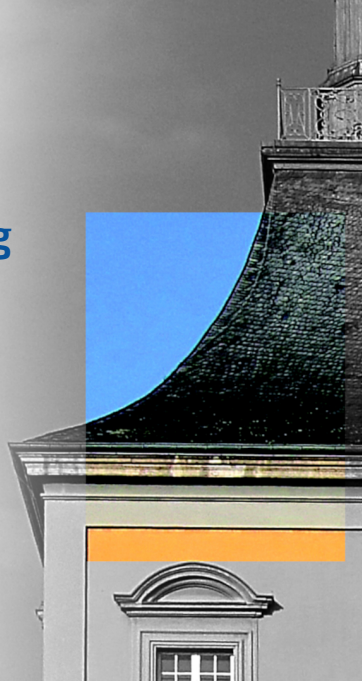
UNIVERSITÄT BONN

Algorithmen und Programmierung

Einleitung

Dr. Felix Jonathan Boes
boes@cs.uni-bonn.de
Institut für Informatik



Thematischer Überblick

In diesem Modul erlernen Sie die Grundlagen der Algorithmik und Programmierung

Die konkret erlernten Programmiersprachen sind unwichtiger als die dahinter stehenden Konzepte und Methoden

In Ihrem Studienverlauf werden Sie selbstständig neue Programmiersprachen erlernen sowie die Konzepte und Methoden wiedererkennen und vertiefen

UNIVERSITÄT BONN

Geplanter Inhalt

- Einleitung
- Motivation: Theorie und Praxis um Sortierverfahren zu studieren
  - Imperative Programmierung (in C++)
  - Algorithmen vollständig beschreiben
  - Speicherverbrauch und Laufzeiten analysieren
- Motivation: Theorie und Praxis um Graphen zu modellieren
  - Graphen und Algorithmen auf Graphen
  - Abstrakte Datentypen und Datenstrukturen
  - Objektorientierte Programmierung (in C++)

Disclaimer

Die Einleitung ist sehr theoretisch

Die Einleitung dient Ihnen als Hintergrundwissen

Lassen Sie sich von der Fülle an Details nicht einschüchtern

In diesem Foliensatz sind die meisten Folien mit der Eule der Minerva gekennzeichnet

Algorithmen

Was sind Algorithmen?

Was sind wesentliche Merkmale von Algorithmen?

UNIVERSITÄT BONN

Einstiegsbeispiel

Sie ziehen nach Bonn und haben 5 Umzugskartons, die sie sortiert nach Gewicht verladen müssen.

```
Input:  5 Umzugskartons mit unbekanntem Gewicht
Output: Dieselben 5 Umzugskartons, sortiert nach Gewicht
Algorithmus (Min-Weight-Sort):
1) Finde den leichtesten Umzugskarton und stelle ihn auf den
   vordersten freien Platz
2) Wiederhole den Schritt mit den verbleibenden Umzugskartons
```

Haben Sie Fragen?

UNIVERSITÄT BONN

Einordnung in den wissenschaftlichen Kontext I

Die wissenschaftlichen Disziplinen verfolgen vielfältige Ziele

- **Mathematik**: Abstrakte Strukturen modellieren und untersuchen
- **Physik**: Naturphänomene untersuchen, modellieren und deren Gesetzmäßigkeiten erklären
- **Informatik**: Informationen darstellen, speichern, verarbeiten und übertragen

UNIVERSITÄT BONN

Einordnung in den wissenschaftlichen Kontext II

Algorithmen sind Kernwerkzeuge und Untersuchungsgegenstände der Informatik

(Algorithmus)

Als **Algorithmus** bezeichnen wir ein eindeutig beschriebenes Verfahren zur Lösung eines klar definierten Problems durch, möglicherweise parallel ausgeführte, Folgen von endlich vielen, präsizen Anweisungen.

Dabei muss angegeben sein:

- Spezifikation von Ein- und Ausgabegrößen
- Spezifikation der zulässigen Anweisungen
- Korrektheit

UNIVERSITÄT BONN

Einordnung in den wissenschaftlichen Kontext III

Überblick über verschiedene Kategorien von Algorithmen

- Sortierverfahren
- Graphenalgorithmen (Z.B. Algorithmen auf Straßennetzen)
- Greedyverfahren
- Probabilistische und Randomisierte Verfahren
- Speicherstrukturen
- Approximative Verfahren
- Lernverfahren
- Onlineverfahren (Algorithmus erhält Daten live / während der Verarbeitung)
- Verteilte Verfahren

Haben Sie Fragen?

Algorithmen

Wesentliche Merkmale

UNIVERSITÄT BONN

Algorithmen: Notation

Algorithmen werden unter anderem wie folgt beschrieben.

- Umgangssprachlich
- Pseudocode
- Konkrete Programmiersprache
- Flussdiagram / Ablaufdiagram

UNIVERSITÄT BONN

Algorithmen: Ein- und Ausgabe

Jeder Algorithmus spezifiziert welche Eigenschaften die Eingabedaten haben. Die Eingabedaten liegen meistens zum Beginn des Verfahrens zur Verfügung.

Jeder Algorithmus spezifiziert welche Eigenschaften die Ausgabedaten haben. Die Ausgabedaten liegen meistens am Ende des Verfahrens vor.

UNIVERSITÄT BONN

Algorithmen: Zulässige Anweisungen

Jeder Algorithmus ist als Folge(n) von zulässigen Anweisungen formuliert. Welche Anweisungen zulässig sind, hängt vom Kontext ab in dem der Algorithmus formuliert wird.

- Bei der umgangssprachlichen Formulierung hängt es vom Gegenüber ab...
- Bei Algorithmen in Pseudocode sind alle Anweisungen zulässig, die von der jeweiligen „Community“ akzeptiert sind...
- Beim Programmieren werden die zulässigen Anweisungen durch die Spezifikation der Programmiersprache festgelegt.

UNIVERSITÄT BONN

Algorithmen: Korrektheit

Jeder Algorithmus ist als Verfahren von Anweisungen beschrieben. Dieses Verfahren **soll** ein gegebenes Problem lösen.

Man sagt dass ein **Algorithmus korrekt** ist, wenn das beschriebene Verfahren das gegebene Problem löst.

Achtung: Es reicht nicht ein Verfahren hinzuschreiben und zu behaupten dass es das gegebene Problem löst. Hier ist es notwendig die Leser:innen durch **Argumente** zu überzeugen.

Algorithmen: Effizienz

Effizienz ist ein Qualitätsmerkmal von Algorithmen und Software.

Jeder Algorithmus kostet. Welche Kosten relevant sind hängt vom Betrachter ab. Beispielsweise werden Sortierverfahren oft in der Anzahl der Vergleiche gemessen. Andere Messgrößen sind zum Beispiel Laufzeit oder Speicherverbrauch.

Die **Effizienz** von Algorithmen ist ein wesentlicher Untersuchungsgegenstand der Informatik.

UNIVERSITÄT BONN

Algorithmen: Lesbarkeit

Lesbarkeit ist ein Qualitätsmerkmal von Algorithmen und Software.

Algorithmen werden von Menschen gelesen und geschrieben. Algorithmen die ihre Kernidee einfach und klar präsentieren erhöhen die Lesbarkeit. Je lesbarer ein Programm ist, desto wertvoller ist es für die Leser:innen. Außerdem sind lesbare Programme weniger anfällig für Implementierungs- oder Denkfehler.

Es ist Ihre Aufgabe gut lesbare Algorithmen und gut lesbaren Programmcode zu schreiben. Es ist nicht die Aufgabe der Leser:innen schlecht aufgeschriebene Algorithmen oder schlecht aufgeschriebenen Programmcode zu verstehen.

Algorithmen: Zielkonflikte

Bei spannenden Problemen ist es fast nie möglich einen Algorithmus zu finden der das allgemeine spannende Problem perfekt löst, leicht zu verstehen ist und effizient arbeitet.

In diesem Fall müssen Kompromisse gemacht werden. Beispielsweise kann es ausreichen nicht „die besten Lösung“ zu ermitteln sondern eine „Lösung die höchstens 0.01% von der besten Lösung abweicht“ effizient zu ermitteln.

Algorithmen: Zielkonflikte

Ein Klassiker ist das **Rundreiseproblem** (Traveling Salesman Problem). Gesucht ist ein Algorithmus der folgendes Problem immer effizient löst.

- Input: Eine Menge von Reisezielen.
- Output: Ein kürzester Rundweg durch alle Reiseziele (natürlich auf dem Rad und nicht per Flugzeug).

Es ist kein Algorithmus bekannt der (immer) einen kürzesten Weg effizient berechnen kann. Aber es sind Algorithmen bekannt die einen kurzen Weg effizient berechnen. Der kurze Weg ist im schlimmsten Fall höchstens 40% länger ist als ein kürzester Weg.

Haben Sie Fragen?

Algorithmen

Abstraktion

Abstraktion I

Bei der Untersuchung und Entwicklung von Algorithmen ist die **Abstraktion** eine ständige Begleiterin. Einfach gesagt ist Abstraktion das Entfernen von unwesentlichen Bestandteilen.

Betrachten wir nochmal unser Sortierbeispiel mit 5 Umzugskartons:

- Dass es genau diese 5 Umzugskartons sind ist unwesentlich.
- Die Gesamtzahl der Umzugskartons ist unwesentlich.
- Dass es Umzugskartons sind ist unwesentlich, es reicht den Umzugskartons Zahlenwerte zuzuweisen.
- Dass es Zahlenwerte sind ist unwesentlich, es reicht dass es (teilweise) vergleichbare Elemente sind.

UNIVERSITÄT BONN

Abstraktion II

Eine abstraktere Version unseres Verfahrens sieht dann so aus:

```
1 Input:  Eine Menge von (teilweise) vergleichbaren Dingen.
2 Output: Eine Sortierung dieser Menge sodass
3         x vor y kommt wenn x kleiner als y ist.
4 Algorithmus (Min-Sort):
5 1) Finde ein Element welches am kleinsten ist und lege es auf den
6    vordersten freien Platz.
7 2) Wiederhole den Schritt mit den verbleibenden Elementen.
```

Augenscheinlicher Vorteil: Wir können nun viel mehr Dinge sortieren.

Augenscheinlicher Nachteil: Wir *wissen nicht genau* was es heißt dass zwei Dinge teilweise vergleichbar sind. Außerdem: Warum ist der Algorithmus korrekt?

Abstraktion III

Neben der Abstraktion eines Verfahrens ist die **Abstraktion des Ausführungsmodells** wesentlich.

Betrachten wir nochmal unser Sortierbeispiel mit 5 Umzugskartons:

- Dass ein Mensch den Algorithmus ausführt ist unwesentlich. Es kann ja auch genau mein Laptop ausführen.
- Dass genau mein Laptop den Algorithmus ausführt ist unwesentlich. Es kann ja auch ein Computer ausführen der eine ähnliche CPU hat.
- Dass ein Computer mit ähnlicher CPU den Algorithmus ausführt ist unwesentlich. Es reicht wenn es ein Ding erledigt was sich so verhält wie ein Computer.

UNIVERSITÄT BONN

Abstraktion IV

Schlussendlich gelangen wir so zu einem abstrakten **Computermodell** auf dem der Algorithmus ausgeführt wird.

Wenn Sie nachlesen welche Befehle eine moderne CPU zur Verfügung stellt, dann stellen Sie fest, dass ein abstraktes Computermodell beschrieben wird welches spezifische Eigenschaften erfüllt.

Bei der Spezifikation von Programmiersprachen ist es ähnlich. Hier wird beschrieben wie sich die Befehle auf einem abstrakten Computermodell verhalten müssen.

Sie werden diesem Umstand an einigen Stellen in dieser Vorlesung und in Ihrem Leben als Informatiker:in begegnen.

Haben Sie Fragen?

Zusammenfassung

Sie haben gelernt was Algorithmen sind

Sie haben wesentliche Merkmale von Algorithmen kennengelernt

Sie haben von Ihrer ständigen Begleiterin der Abstraktion erfahren

Programmierung

Offene Fragen

Was ist ein Computerprogramm?

Wo und wie wird ein Computerprogramm ausgeführt?

Welche Programmiersprachen lernen wir?

Programmierung

Modelle

UNIVERSITÄT BONN

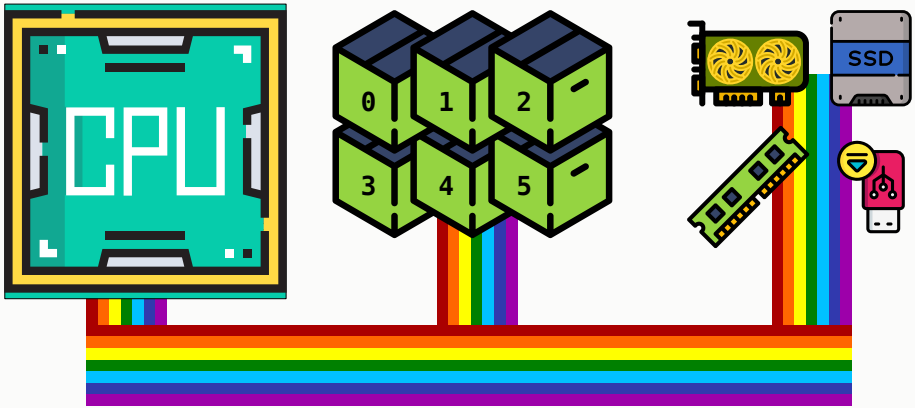
Computermodell

Es gibt unterschiedlich abstrakte Modelle für Computer. In der theoretischen Informatik verwendet man oft das Modell der **Turingmaschine**. Das ist ein unendlich langes Band mit einer Einheit die das Band ließt und beschreibt.

Hier verwenden wir zunächst nur unser „intuitives“ Computermodell: Es gibt eine oder mehrere CPUs die alle Programmbefehle verarbeiten und eine Menge von Speicheradressen um Daten abzulegen.

UNIVERSITÄT BONN

Unser einfaches Computermodell



Es gibt eine oder mehrere CPUs und eine Menge von Speicheradressen um Daten abzulegen. Außerdem gibt es Möglichkeiten auf weitere Ressourcen zuzugreifen.

UNIVERSITÄT BONN

Programme und Prozesse

Ein **Computerprogramm** ist eine Folge von Anweisungen die in der Sprache des Computermodells geschrieben ist. Ein **Prozess** ist ein in der Ausführung befindliches Programm.

Einem **Betriebssystemprozess** stehen alle priviligierten und unpriviligierten Anweisungen der CPUs zur Verfügung. Einem **Nutzerprozess** stehen alle unpriviligierten Anweisungen zur Verfügung. Außerdem steht jedem Nutzerprozess ein exklusiver **virtueller Adressraum** zur Verfügung. Wie dieser virtuelle Adressraum mit dem physischen Speicher (RAM) zusammenhängt lernen Sie später.

UNIVERSITÄT BONN

Programmiersprache

Eine **Programmiersprache** ist eine Sprache um Computerprogramme zu schreiben. Dabei wird unter anderem festgelegt, was die **grundlegenden Datentypen** sind. Die Programmiersprache legt fest wie die grundlegenden Datentypen verändert werden und wie sich das Computermodell bei der Ausführung des Programms verhalten muss.

UNIVERSITÄT BONN

Grundlegende Datentypen
C, C++, Java und weitere

In den Programmiersprachen C, C++, Java und vielen weiteren werden folgende grundlegende Datentypen definiert.

Datentyp	Beschreibung	min	max
`int8`	8-bit Ganzzahl	-128	127
`int16`	16-bit Ganzzahl	-32.768	32.767
`int32`	32-bit Ganzzahl	$\approx$ -2,1 Milliarden	$\approx$ 2,1 Milliarden
`int64`	64-bit Ganzzahl	$\approx$ -9,2 Trillionen	$\approx$ 9,2 Trillionen
`uint8`	8-bit natürliche Zahl	0	255
`...`	...	...	...
`double`	64-bit Gleitkommazahl	$\approx -1.8 \cdot 10^{308}$	$\approx 1.8 \cdot 10^{308}$
`char`	8-bit Zeichen	Nicht druckbar	Gewisses Leerzeichen
`wchar`	16-bit oder 32-bit Zeichen	Nicht druckbar	Hängt davon ab...

Grundlagen Datentypen
Python

Python bietet andere grundlegende Datentypen.

Datentyp	Beschreibung	Weiteres
`int`	Ganzzahl	Größe beliebig
`float`	64-bit Gleitkommazahl	$\approx$ -1.8 $\cdot 10^{308}$ bis $\approx$ 1.8 $\cdot 10^{308}$
`complex`	komplexe Gleitkommazahl	also $z = x + \sqrt{-1} \cdot y$
`str`	Text	
`list`	Veränderbare Sequenz von Objekten	
`...`	...	...

Sie lernen mehr zu diesem Kontext im Modul Technische Informatik und Weiteren

Fragen?

Programmierung

Programmiersprachen

UNIVERSITÄT BONN

Beispiel Python

So sieht ein „Hallo Welt Programm“ in Python aus.

```
def main():
  print('Hallöchen')

if __name__ == '__main__':
  main()
```

So wird das Programm ausgeführt.

```
python3 hallo_welt.py
```

Beispiel x64 Assembler

So sieht ein „Hallo Welt Programm“ in x64 Assember aus.

```
section .text
global _start
_start:
  mov rax, 1
  mov rdi, 1
  mov rsi, msg
  mov rdx, msglen
  syscall

  ...

section .rodata
  msg: db "Hallöchen", 10
  msglen: equ $ - msg
```

So wird das Programm ausgeführt.

```
nasm -f elf64 hallo_welt.asm
ld -o hallo_welt hallo_welt.o
./hallo_welt
```

Beispiel C++17

So sieht ein einfaches „Hallo Welt Programm“ in C++17 aus.

```
#include <iostream>

int main() {
  std::cout << "Hallöchen" << std::endl;
  return 0;
}
```

So wird das Programm ausgeführt.

```
clang++ -std=c++17 -o hallo_welt hallo_welt.cpp
./hallo_welt
```

C++17 imperativ und objektorientiert

In diesem Modul lernen Sie zuerst Konzepte der imperativen (und klassenlosen) Programmierung in C++17 kennen. Im Anschluss lernen Sie die Konzepte der objektorientierten Programmierung in C++17 kennen.

Haben Sie Fragen?

Zusammenfassung

Sie haben etwas über Computermodelle und Programme gelernt

Sie haben Beispiele für Programmiersprachen gesehen

Sie wissen jetzt dass Sie C++ lernen werden