Teil I

Einführung

Überblick

Was ist Informatik?

- Informatik hat zentral zu tun mit
  - systematischer Verarbeitung von Informationen
  - Maschinen, die diese Verarbeitung automatisch leisten ($\longrightarrow$ Computer)

Die „systematische Verarbeitung“ wird durch den Begriff Algorithmus präzisiert, Information durch den Begriff Daten.

Was ist ein Algorithmus?

- allgemein:

Ein Algorithmus ist eine eindeutige Beschreibung eines in mehreren Schritten durchgeführten (Bearbeitungs-)Vorganges.

- in der Informatik speziell: Berechnungsvorgänge statt Bearbeitungsvorgänge, Schwerpunkt auf *Ausführbarkeit* durch (abstrakte) Maschinen.

Ein *Prozessor* führt einen Prozess (Arbeitsvorgang) auf Basis einer eindeutig interpretierbaren Beschreibung (dem Algorithmus) aus.

Typische algorithmisierbare Prozesse

- Kochrezepte
  - Pizza aufwärmen
  - Seeteufel mit Kräuterkruste auf Lauch
- Bedienungsanleitungen
  - Suchen eines Telefonbucheintrags
  - Herausgeben von Wechselgeld
  - Sortieren von Spielkarten
- Berechnungsvorschriften
  - schriftliches Addieren
  - Berechnung der Fakultät ($x! = x \cdot (x-1) \cdot \cdots \cdot 3 \cdot 2 \cdot 1$)
  - Berechnung des größten gemeinsamen Teilers

Algorithmus-Begriff

Ein Algorithmus ist eine präzise (in einer festgelegten Sprache abgefasste) endliche Beschreibung eines allgemeinen Verfahrens unter Verwendung ausführbarer elementarer (Verarbeitungs-)Schritte.

Ein Programm ist die Formulierung eines Algorithmus in einer konkreten Programmiersprache.

Bekannte Algorithmen

1. Addition zweier positiver Dezimalzahlen (mit Überträgen)

$$\begin{array}{rcc} & 3 & 3 \\ + & 4 & 8 \\ \hline & 1 & \\ \hline & 8 & 1 \end{array}$$

2. Test, ob eine gegebene natürliche Zahl eine Primzahl ist
3. Sortieren einer unsortierten Kartei (etwa lexikographisch)
4. Berechnung der Zahl $e = 2.7182\ldots$

Eigenschaften von Algorithmen

- Notation für Beschreibung
- Ausdrucksfähigkeit
- Terminierung
- Determinismus
- Berechenbarkeit
- Korrektheit / Genauigkeit / Eindeutigkeit
- Zeitbedarf / Geschwindigkeit

Notation

"Hal, bitte berechne mir die Fakultät von 20!"

$$0! = 1$$
$$x! = x \cdot (x-1)!$$

```
x = 20; y = 1;
while (x > 1) {
    y = y * x;
    x = x - 1;
}
return y;
```

Ausdrucksfähigkeit

- verschiedene Notationen = gleiche Ausdrucksfähigkeit?
- Wahl der Programmiersprache – universelle Programmiersprache?
- aber, man vergleiche z.B.
  - Bienensprache vs. Sprache zur Wegfindungsprogrammierung für Roboter

Terminierung

Ein Algorithmus heißt terminierend, wenn er (bei jeder erlaubten Eingabe von Parameterwerten) nach endlich vielen Schritten abbricht.

- **Frage:** Terminieren die Algorithmen von Folie 2-3?
- Beispiel

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \cdots = \sum_{n=0}^{\infty} \frac{1}{n!}$$

Determinismus

- legt „Wahlfreiheit“ bei der Ausführung fest
- Formen
  - **deterministischer Ablauf**: eindeutige Vorgabe der Folge der auszuführenden Schritte
  - **determiniertes Ergebnis**: eindeutiges Ergebnis bei vorgegebenen Parameterwerten

Determinismus /2

- Nichtdeterminierter vs. determinierter Algorithmus
- Beispiel #1

1. Nimm eine beliebige Zahl x
2. Addiere 5 hinzu und multipliziere mit 3
3. Gib das Ergebnis aus

- Beispiel #2

1. Nimm eine Zahl $x \neq 0$
2. Entweder: Addiere das Dreifache von x zu x und teile das Ergebnis durch x: $(3x + x)/x$
   Oder: Subtrahiere 4 von x und subtrahiere das Ergebnis von x: $x - (x - 4)$
3. Gib das Ergebnis aus

Berechenbarkeit

- Kann man „alles“ programmieren (berechnen)?

- Antwort: Nein!
- Nichtentscheidbare Probleme
  - Halteproblem
  - semantische Eigenschaften von Algorithmen, z.B.:
    - Berechnen zwei Algorithmen dieselbe Funktion?
    - Ist ein gegebener Algorithmus korrekt, d.h. berechnet er die gegebene (gewünschte) Funktion?

Korrektheit

- Algorithmen (Programme) sollen sich wie beabsichtigt verhalten
  $\rightsquigarrow$ Korrektheit
- Beispiele für Programmfehler
  (`de.wikipedia.org/wiki/Programmfehler`)
  - 1962: Verlust der Venus-Sonde Mariner 1 durch fehlenden Bindestrich in einem Fortran-Programm (80 Mill. Dollar)
  - 1996: Zerstörung der Ariane 5-Rakete kurz nach dem Start durch übernommenen und nicht korrekt spezifizierten Programmcode
  - 1999: Verlust der Mars-Sonde Climate Orbiter durch falsches Maßsystem (Yard statt Meter) bei Programmierung

Zeitbedarf

- Wie aufwändig ist ein gegebener Algorithmus?
- Abschätzung des Aufwands
  - unabhängig von konkreter Hardwareleistung (Mooresches Gesetz: „Komplexität von integrierten Schaltungen / Rechenleistung verdoppelt sich alle 18 Monate“)
  - in Abhängigkeit von der Problemgröße
  - für den schlechtesten Fall (worst case) und im Mittel
- Beispiel:

Operation	Aufwand	$n = 2$	$n = 2^{10}$	$n = 2^{20}$
sequenzielle Suche	n	2	1024	1048576
Baum-Suche	$\log n$	1	10	20

Historischer Überblick: Algorithmen

- 300 v. Chr.: Euklids Algorithmus zur Bestimmung des **ggT** (7. Buch der Elemente):

$$\mathbf{ggT}(300, 200) = 100$$

- 800 n. Chr.: Muhammed ibn Musa abu Djafar alChoresmi: Aufgabensammlung für Kaufleute und Testamentsvollstrecker (lat.: Liber Algorithmi, Kunstwort aus dem Namen und griechisch „arithmos“ für Zahl)
- 1574: Adam Rieses Rechenbuch
- 1614: Logarithmentafeln (30 Jahre für Berechnung!)
- 1703: Binäres Zahlensystem (Leibnitz)
- 1931: Gödels Unvollständigkeitssatz
- 1936: Church’sche These