Hardwarenahe Programmierung I

U. Kaiser, R. Kaiser, M. Stöttinger, S. Reith

(HTTP: http://www.cs.hs-rm.de/~kaiser
EMail: robert.kaiser@hs-rm.de)

Wintersemester 2021/2022

2. Einige Grundbegriffe

http://www.denkschatz.de/zitate/Edsger-W-Dijkstra/In-der-Informatik-geht-es-genau-so-wenig-um-Computer-wie-in-der-Astronomie

Ein Rezept

Hochschule RheinMain

Määnzer Spundekäs[1]

250 g	Butter
600 g	Frischkäse
750 g	Quark, (40% Fett)
2	Knoblauchzehe(n)
5 TL	Paprikapulver, edelsüß
	Salz und Pfeffer
$\frac{1}{2}$	Zwiebel(n), zum Garnieren

Zubereitung

Butter auf Zimmertemperatur mit dem Frischkäse und Quark mischen, geht am besten mit einem Handrührgerät (ganz wichtig - nicht zu stark und nicht zu lange rühren). Nur bis die Zutaten sich vermischt haben, dann sollte man mit einem Schneebesen weiter rühren. Wenn man zu lange rührt, wird der Spundekäs matschig (im Original wird der Spundekäs nur mit einem Kochlöffel oder Schneebesen gerührt, das dauert aber seine Zeit, deshalb kann man auch mal den Rührer benutzen, wenn man nicht allzu viel Zeit hat).
Jetzt den Knoblauch schälen und klein hacken und unter die Masse heben. Zum Schluss mit Salz, Pfeffer und dem Paprikapulver würzen. Der Spundekäs sollte eine leicht hellrote Farbe haben.
Kurz bevor der Spundekäse gegessen wird, kann man zum Garnieren eine halbe Zwiebel schneiden und über den Spundekäse streuen.
Ganz wichtig: im Sommer wirklich kurz vor dem Essen die Zwiebel darüber machen sonst kann die Masse kippen. Am besten schmeckt der Spundekäse, wenn er 1 Nacht im Kühlschrank durchgezogen ist.
Mit kleinen Salzbrezeln oder auch frischen Laugenbrezeln kann man den Spundekäse essen. Lecker dazu ist auch einfach ein frisches Baguette zum Dippen.

[1] `https://www.chefkoch.de/rezepte/1998211323695447/Meenzer-Spundekaes.html`

Analogien

Hochschule RheinMain

- Die Zutaten für das Rezept sind die **Daten** bzw. **Datenstrukturen**, die wir verarbeiten wollen.
- Die Zubereitungsvorschrift ist ein **Algorithmus**, der festlegt, wie die Daten zu verarbeiten sind.
- Das Rezept insgesamt ist ein **Programm**, das alle Datenstrukturen (Zutaten) und Algorithmen (Zubereitungsvorschriften) zum Lösen der gestellten Aufgabe enthält.
- Die gemeinsame Terminologie, in der sich Autor und Leser des Rezepts verständigen, ist die **Programmiersprache**, in der das Programm geschrieben ist. Die Programmiersprache muss dabei alle bezüglich der Zutaten und der Zubereitung bedeutsamen Informationen zweifelsfrei zu übermitteln.
- Die Küche ist die technische Infrastruktur zur Umsetzung von Rezepten in schmackhafte Gerichte und ist vergleichbar mit einem **Computer**, seinem **Betriebssystem** und den benötigten **Entwicklungswerkzeugen**.
- Der Koch übersetzt das Rezept in einzelne Arbeitsschritte in der Küche. Üblicherweise geht ein Koch in zwei Schritten vor. Im ersten Schritt bereitet er die Zutaten einzeln und unabhängig voneinander vor (z. B. Kartoffeln kochen), um die Einzelteile dann in einem zweiten Schritt zusammenzufügen und abzuschmecken. In der Datenverarbeitung sprechen wir in diesem Zusammenhang von **Compiler** und **Linker**.
- Das fertige Gericht ist das **lauffähige Programm**, das vom Benutzer (Esser) angewandt (verzehrt) werden kann.

Algorithmus

Hochschule RheinMain

- Ein **Algorithmus** ist eine endliche Menge von genau beschriebenen Anweisungen, die unter Benutzung von vorgegebenen Anfangsdaten in einer genau festgelegten Reihenfolge auszuführen sind, um die Lösung eines Problems in endlich vielen Schritten zu ermitteln.
- Beispiel: Der Algorithmus zur schriftlichen Division

```
84 : 16 = 5,25
80
---
 40
 32
 ---
  80
  80
  --
   0
```

Beispiel

Hochschule RheinMain

Problem:
Berechne den Quotienten zweier natürlicher Zahlen!
Anfangsdaten:
$z =$ Zähler($z \geq 0$),
$n =$ Nenner($n > 0$) und
$a =$Anzahl der zu berechnenden Nachkommastellen.
Anweisungen:

1. Bestimme die größte ganze Zahl x mit $n \cdot x \leq z$!
   Dies ist der Vorkomma-Anteil der gesuchten Zahl.
2. Zur Bestimmung der Nachkommastellen fahre wie folgt fort:

2.1 Sind noch Nachkommastellen zu berechnen (d. h. $a > 0$)?
Wenn nein, dann beende das Verfahren!

2.2 Setze $z = 10(z - nx)$!

2.4 Ist $z = 0$, so beende das Verfahren!

2.5 Bestimme die größte ganze Zahl x mit $nx \leq z$!
Dies ist die nächste Ziffer.

2.6 Jetzt ist eine Ziffer weniger zu bestimmen.
Vermindere also den Wert von a um 1 und fahre anschließend bei 2.1 fort!

```
84 : 16 = 5,25
80
---
 40
 32
 ---
  80
  80
  --
   0
```

Anfänglich ist a die Anzahl der zu berechnenden Nachkommastellen. Im Verfahren verwenden wir a als die Anzahl der noch zu berechnenden Nachkommastellen. Wir werden den Wert von a in jedem Verfahrensschritt herunterzählen, bis $a = 0$ ist und keine Nachkommastellen mehr zu berechnen sind.

Flussdiagramm

Hochschule RheinMain

1. Bestimme die größte ganze Zahl x mit $n \cdot x \leq z$! Dies ist der Vorkomma-Anteil der gesuchten Zahl.
2. Zur Bestimmung der Nachkommastellen fahre wie folgt fort:

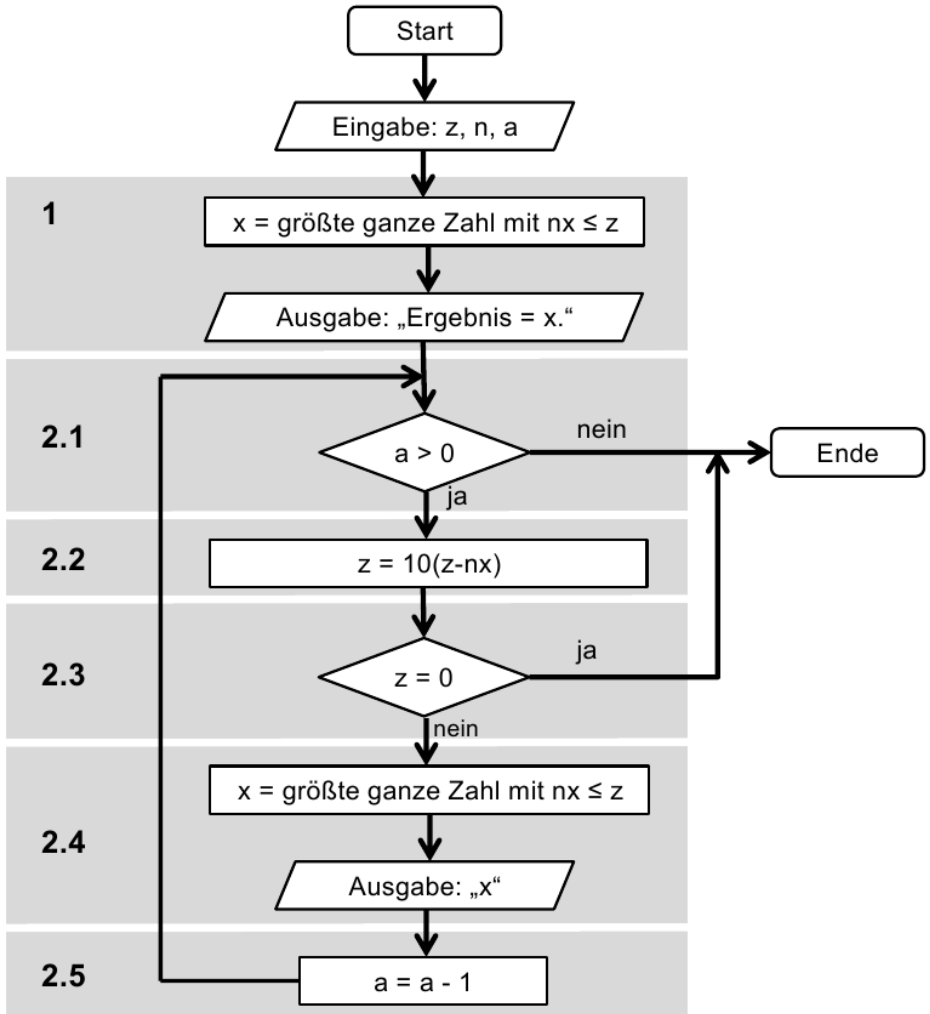
2.1 Sind noch Nachkommastellen zu berechnen (d. h. $a > 0$)? Wenn nein, dann beende das Verfahren!

2.2 Setze $z = 10(z - nx)$!

2.4 Ist $z = 0$, so beende das Verfahren!

2.5 Bestimme die größte ganze Zahl x mit $nx \leq z$! Dies ist die nächste Ziffer.

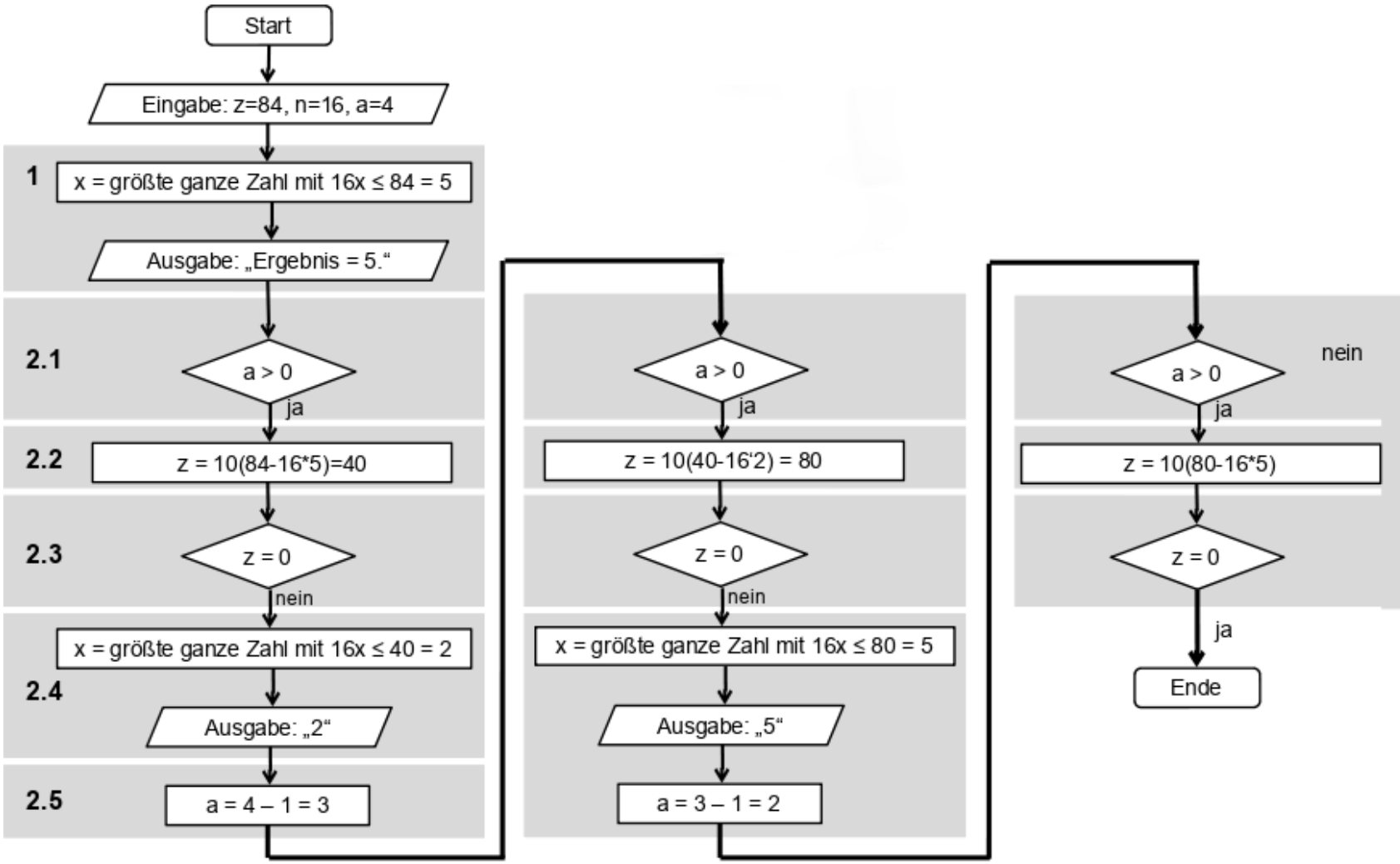
2.6 Jetzt ist eine Ziffer weniger zu bestimmen. Vermindere also den Wert von a um 1 und fahre anschließend bei 2.1 fort!



Konkreter Ablauf

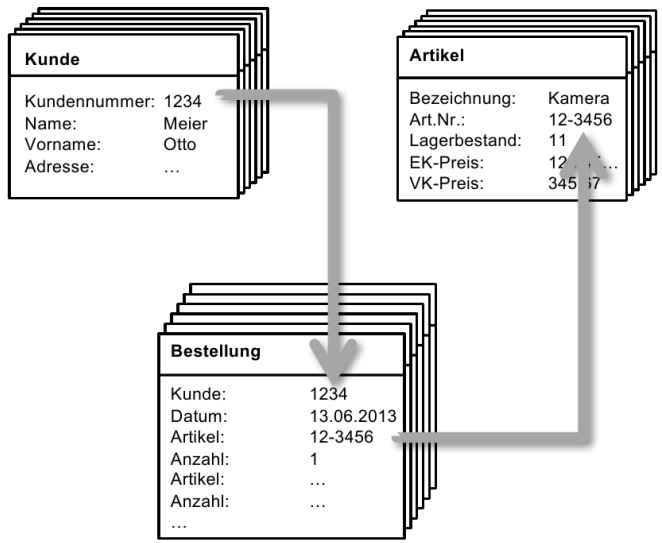
Hochschule RheinMain

$$84 : 16 = 5,25$$



Datenstruktur

- Eine **Datenstruktur** ist ein Modell, das die zur Lösung eines Problems benötigten Informationen (Ausgangsdaten, Zwischenergebnisse, Endergebnisse) aufnimmt und für alle Informationen genau festgelegte Zugriffswege bereitstellt.
- Beispiel: Karteikästen eines Versandhandels



- So wie es sich bei einem *Wassereimer* nicht um *Wasser* handelt, handelt es sich bei einer *Datenstruktur* nicht um *Daten*. Die Datenstruktur ist nur ein *Behältnis* für Daten.
- Technisch realisiert werden Datenstrukturen mit Hilfe von *Speicher*.
- Der Inhalt der Datenstruktur (d.h. die Daten) kann im Laufe der Ausführung eines Algorithmus verändert werden.

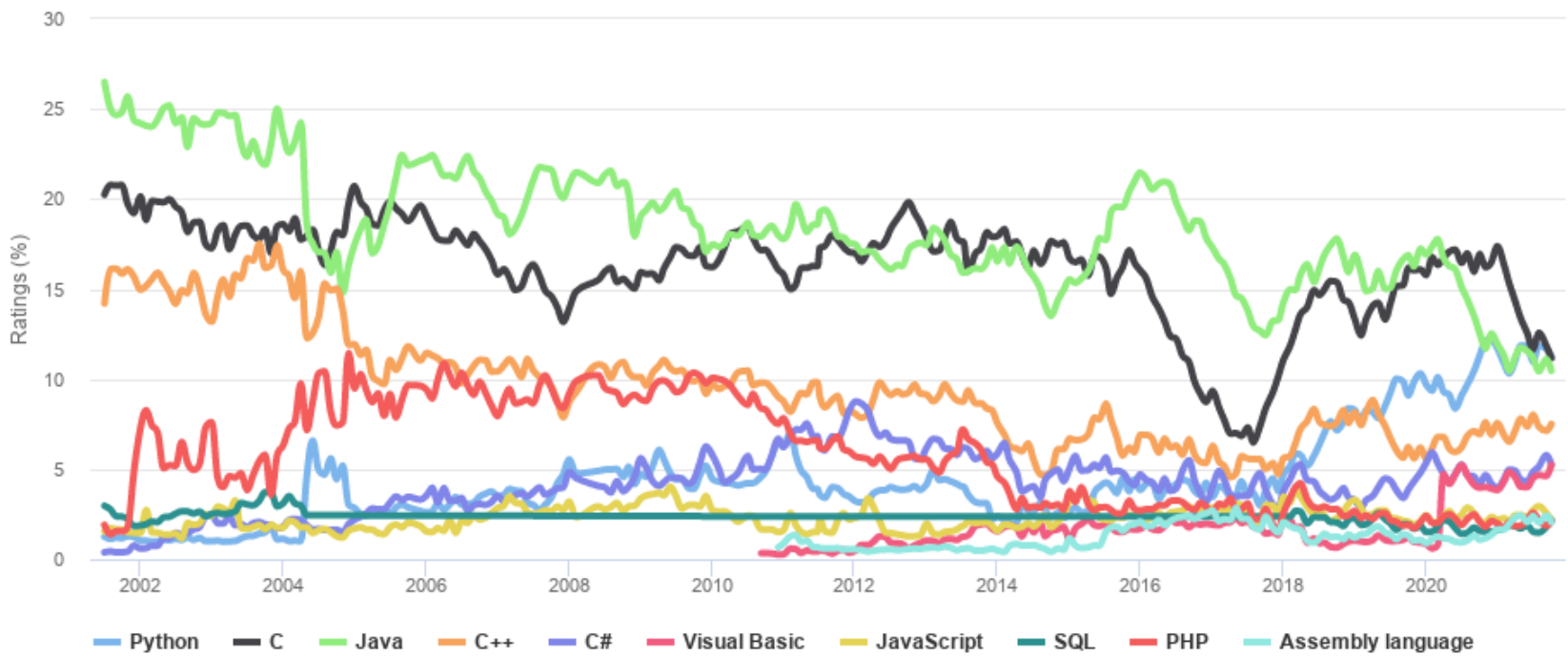
Programm

- Ein **Programm** ist eine eindeutige, formalisierte Beschreibung von Algorithmen und Datenstrukturen, die durch einen automatischen Übersetzungsprozess auf einem Computer ablauffähig ist.
- Auch Programme können grundsätzlich wie Daten gespeichert werden.

Programmiersprachen

Hochschule RheinMain

- Den zur Formulierung eines Programms verwendeten Beschreibungsformalismus bezeichnen wir als **Programmiersprache**.
- Entwicklung der Verbreitung verschiedener populärer Programmiersprachen der letzten 20 Jahre (Tiobe Index[2]):



[2] https://www.tiobe.com/tiobe-index/