de Gruyter Lehrbuch
Hommel · Jähnichen · Koster
Methodisches Programmieren

Günter Hommel
Stefan Jähnichen
Cornelis H. A. Koster

Methodisches Programmieren

Entwicklung von Algorithmen durch schrittweise Verfeinerung

Walter de Gruyter · Berlin · New York 1983

Die Autoren

Dr. Günter Hommel
Professor für Informatik
an der Technischen Universität München

Dr. Stefan Jähnichen
Hochschulassistent am Fachbereich Informatik
an der Technischen Universität Berlin

Cornelis H. A. Koster
Professor für Informatik
an der Katholieke Universiteit Nijmegen

CIP-Kurztitelaufnahme der Deutschen Bibliothek

Hommel, Günter:
Methodisches Programmieren:
Entwicklung von Algorithmen durch schrittweise Verfeinerung/
Günter Hommel; Stefan Jähnichen; Cornelis H. A. Koster. –
Berlin; New York: de Gruyter, 1983.
(de Gruyter-Lehrbuch)
ISBN 3-11-009636-6
NE: Jähnichen, Stefan; Koster, Cornelis H.A.:

Druck: Gerike, Berlin. Bindearbeiten: Lüderitz & Bauer, Buchgewerbe GmbH, Berlin.
Printed in Germany.

Vorwort

Dieses Buch ist eine Einführung in die Algorithmik. Es wendet sich an Anfänger der Informatik, sowohl an Universitäten und Fachhochschulen als auch an Pädagogischen Hochschulen, kann aber auch im Informatikunterricht an allgemeinbildenden Schulen und Berufsschulen benutzt werden. Nicht zuletzt möchten wir mit diesem Buch eine Generation von Lehrern ansprechen, die im Stande ist, die beschriebene Methodik im Schulgebrauch zu verwerten.

Das systematische Entwerfen und Realisieren von Algorithmen ist das zentrale Anliegen dieser Einführung. Als Hilfsmittel zur Formulierung von Algorithmen wird die Programmiersprache ELAN benutzt [Hommel 79], [Kleine 81]. Der in diesem Buch verfolgte didaktische Ansatz kann jedoch auch bei Verwendung anderer höherer Programmiersprachen (PASCAL oder andere Sprachen der ALGOL-Familie) beibehalten werden.

Der Zugang zur Algorithmik ist bewusst sehr intuitiv und wenig formal gehalten, was nach unseren Erfahrungen der Verständlichkeit und Einsicht zugute kommt. Für die Ausbildung von Studenten, die Informatik nicht als Hauptfach wählen, ist der Umfang des vorliegenden Buches eine ausreichende Einführung in die Algorithmik. Für Informatikstudenten müssen daneben auch die formalen Konstruktionsmethoden gelehrt werden. Auch dann behält aber der intuitive Zugang seinen Stellenwert.

Das Buch vermittelt keine Lehrinhalte, die beziehungslos nebeneinander stehen und kritiklos auswendig gelernt werden sollen. Vielemehr wird versucht, Kenntnisse und Fähigkeiten beim Leser zu entwickeln, die für eine systematische Programmierung unerlässlich sind. Neben dem Studium dieses Buches ist praktisches Üben unverzichtbar zum Erwerb der notwendigen Fertigkeiten. Ausserdem können nur durch eine direkte Konfrontation mit einer Rechenanlage die Anwendbarkeit und Grenzen der angebotenen Methoden empfunden werden.

Das Buch versucht im Hinblick auf die Anwendungsgebiete der Informatik neutral zu sein. Mit Absicht sind nur wenige der Beispiele aus den mathematischen Anwendungen gewählt, obwohl gerade dort viele interessante und sinnvolle Aufgaben zu finden sind. Eine Orientierung auf ein bestimmtes Anwendungsgebiet wird in zugehörigen Übungen erfolgen müssen. Beim Leser werden lediglich elementare mathematische Kenntnisse vorausgesetzt.

Zur Programmiermethodik: In diesem Buch wird ausschliesslich die Methode der schrittweisen Verfeinerung (Top-down-Programmierung) vermittelt. In einem folgendem Band werden Methoden zum Entwurf und zur Realisierung grösserer Programme mit den Mitteln der algorithmischen Abstraktion und der Datenabstraktion behandelt. Dazu werden wir uns mehr als im vorliegenden ersten Band auch formaler Konstruktionsmittel bedienen. Diese vielleicht auf den ersten Blick etwas dogmatische Haltung wird durch eine unterschiedliche Zielsetzung der beiden Bände motiviert. Der erste Teil allein ist eine ausreichende Einführung für diejenigen, die in ihrem Fachgebiet Rechner anwenden, aber nur in geringem Umfang selbst programmieren. Dafür ist die hierarchische Dekomposition eine angemessene Methode. Der zweite Teil legt mehr Nachdruck auf die Konstruktion grosser Systeme mit stärkeren Abstraktionselementen, wobei die Beherrschung der Top-down Programmierung vorausgesetzt wird.

Beide Bände zusammen führen zu einem Kenntnisniveau, das für den sogenannten "Soloprogrammierer" typisch ist. Hierauf aufbauend können die Fertigkeiten der "Softwaretechnik", das Entwerfen und Realisieren grösserer und anspruchsvollerer Programme in einem Programmiererteam, unterrichtet werden.

Das Buch ist als Skriptum zu den Vorlesungen "Algorithmen I und II" an der Technischen Universität Berlin entstanden. Im Laufe mehrerer Jahre ist es von vielen Mitarbeitern der Technischen Universität Berlin und später auch der Katholischen Universität Nijmegen beeinflusst worden. Hier soll eine Liste von Personen aufgeführt werden, die sich im Laufe der Jahre mit der Unterrichtsmethodik und mit der Programmiersprache ELAN beschäftigt haben: Wilfried Koch, Karl Kleine, Jochen Jäckel, Eberhard Wegner, Werner Simonsmeier, Jürgen Gottschalk, Thijs Zoethout, Hans Meijer und viele mehr.

Ausserdem müssen Rainer Hahn und Jochen Liedtke von der Universität Bielefeld genannt werden, die sowohl an den Sprachdiskussionen intensiv beteiligt waren, als auch die erste ELAN-Implementierung vorlegen konnten. Anregungen und tatkräftige Unterstützung durch Peter Heyderhoff von der GMD waren hilfreich für die Entwicklung und Implementierung der Sprache.

Besonderer Dank gebührt Frau Gabi Ambach und Frau Josė Aben für das Schreiben des Manuskriptes während der Bearbeitung in immer wieder neuen verbesserten Versionen.

Ganz besonders sind wir aber Karl Kleine verpflichtet. Ohne seine Mitarbeit im Endspurt und ohne sein Insistieren auf dem "jetzt muss es werden" würden wir wahrscheinlich noch immer in alten Listings korrigieren.

Im Januar 1983

G.Hommel / S.Jähnichen / C.H.A.Koster

Inhaltsverzeichnis

1. Einführung in die Informatik

Dieses Kapitel gibt einen groben Überblick über die Wissenschaft Informatik, ihre Entstehungsgeschichte, ihre wichtigsten Themenbereiche und deren Wechselwirkung. Als grundlegende wissenschaftliche Methode zur Lösung komplexer Probleme der Informatik wird die Algorithmik eingeführt.

1.1 Entstehung der Informatik

Wir können hier keine Übersicht der ganzen Entstehungsgeschichte der Informatik geben, sondern nur einige Elemente hervorheben, die zum Begriff der Informatik beigetragen haben. Eine vollständigere historische Übersicht findet man u.a. in [Claus 75] und [Bauer 71].

Auch werden wir nicht versuchen, die Einflüsse der Informatik auf die Gesellschaft, der Gesellschaft auf die Informatik und die Rolle des Informatikers dabei zu beschreiben. Hierzu verweisen wir z.B. auf [Rothman 76] und [Weizenbaum 78].

Das Wort "Informatik" ist eine Neuschöpfung, die etwa 1970 im deutschen und französischen ("informatique") Sprachraum aufkam. In der Anfangsphase ihrer historischen Entwicklung hatten sich die Wissenschafsgebiete, die man heute unter dem Namen "Informatik" zusammenfasst, nicht gleich zu einer zusammengehörigen, eigenständigen Wissenschaft entwickelt, sondern wurden je nach wissenschaftlicher Herkunft und nach Eigenart des Problems als Randgebiet der Mathematik oder der Elektrotechnik begriffen. Die Begegnung fand da statt, wo Mathematiker und Elektrotechniker sich gemeinsam über des neue Wunderding Computer beugten und sich fragten, was man nicht alles "Schönes" damit machen und wie man das Gerät noch schneller, billiger und brauchbarer machen könnte.

In den USA, wo die Entwicklung dieses neuen Gebiets am frühesten vorangetrieben wurde, drückte sich diese Dualität auch in der Namensgebung aus: das Department of Mathematics betrieb "Computing Science", das Department of Electrical Engineering "Computer Science"! Auch in

späteren Auffassungen über Informatik lassen sich diese zwei Gesichtspunkte separat wiederfinden. H. Langmaack beginnt sein Buch [Kandzia 73] mit den Worten: "Die Informatik ... lässt sich charakterisieren als die Lehre vom Aufbau und Einsatz elektronischer Rechenanlagen".

Die Informatik war somit ein Sammelsurium von Techniken und Tricks aus verschiedenen Wissenschaftsgebieten und aus der Praxis, die bestenfalls als eine empirische Wissenschaft zu sehen war. Noch 1969 sah D. Knuth sich veranlasst, sein Standardwerk über die Methoden der Informatik mit "The Art of Computer Programming" zu betiteln [Knuth], also eher Kunstfertigkeit als Wissenschaft. Nicht zuletzt unter dem Einfluss dieses Buches begriff sich etwa ab 1971 die Informatik als eine eigenständige Wissenschaft, mit eigener Methodik, abgrenzbarem Untersuchungsgebiet und Ergebnissen, die durchaus der Strenge und Tiefe nach den Namen einer Wissenschaft verdienen. In Deutschland wird dieser Punkt vom Buch "Informatik" von F.L. Bauer und G. Goos [Bauer 71] markiert. Die Wandlung zu einer theoretisch fundierten Wissenschaft wird auch deutlich durch Titel und Ansatz des Buches "The Science of Programming" [Gries 81] ausgedrückt.

Während der Aufbauphase der Wissenschaft Informatik haben sich vier Teilgebiete herauskristallisiert:

- die technische Informatik,
- die Kerninformatik,
- die theoretische Informatik und
- die angewandte Informatik.

Der Aufbau einer Rechenanlage mit vorhandenen technologischen Mitteln nach einer vorgegebenen Spezifikation, in Abhängigkeit von Preis, Geschwindigheit und Zuverlässigkeit, ist das Gebiet der technischen Informatik.

Der Einsatz vorhandener Rechenanlagen mit vorgegebenen Eigenschaften in Produktion, Verwaltung, Wissenschaft usw., wobei der Nachdruck auf der Lösung existierender Probleme mit wohletablierten Methoden in kürzester Zeit liegt, ist das Gebiet der angewandten Informatik, bei

weitem das grösste Teilgebiet. Dazu gehören typische Anwendungsmethoden der Informatik (z.B. Computer Graphics, Simulationstechniken, Prozesslenkung) im Gegensatz zu den Anwendungsgebieten selbst (Medizin, betriebliche und öffentliche Verwaltung usw.). Der angewandte Informatiker wird häufig in einem Team mit Fachleuten anderer Anwendungsgebietes zusammenarbeiten.

Zwischen diesen beiden Fachgebieten befindet sich die Kerninformatik, die sich mit der Überbrückung der Kluft zwischen den angebotenen Geräten und ihren Anwendungen beschäftigt. Einerseits müssen die Geräte so mit Systemprogrammen "verzuckert" werden, dass die Anwendungen möglichst einfach realisiert werden können, anderseits muss dem technischen Informatiker klargemacht werden, welche Eigenschaften die Maschinen haben müssen, damit die Anwendungen effizient realisiert werden können.

Bis heute lässt die Kommunikation zwischen der Kerninformatik und der technischen Informatik zu wünschen übrig, so dass beim Entwurf der meisten Rechner die technischen Möglichkeiten eine wichtigere Rolle spielen, als die Angemessenheit für die Anwendung. Der Kerninformatiker muss sich meist damit abfinden, das Vorhandene nachträglich zu verzuckern.

Neuere Entwicklungen bei der Konstruktion von Rechnern, insbesondere die Anwendung der Mikroprogrammierung tragen dazu bei, dass die Grenze zwischen technischer Informatik und Kerninformatik immer mehr verwischt wird.

Die Kluft zwischen Maschine und Anwendung bleibt aber bestehen. Man spricht von der "Software-Krise": Die Programmsysteme sind traditionell zu spät fertig, zu teuer, nicht der Spezifikation entsprechend und überdies voller Fehler, die häufig erst während der Anwendung gefunden und beseitigt werden.

Die theoretische Informatik beschäftigt sich mit den Grundlagen der Informatik. Sie untersucht die intuitiv gewonnenen Methoden und versucht mathematische Theorien und Methoden für die Informatik zu entwickeln. Die Mathematik spielt für die Informatik sicherlich eine ebenso grosse Schlüsselrolle wie auch für andere Ingenieurwissenschaften. Trotzdem sind es noch immer meist

die technischen, pragmatischen Methoden, die der theoretischen Erfassung voraneilen und der Informatik einen stark ingenieurmässigen Charakter verleihen und nicht zuletzt dazu beigetragen haben, dass die Datenverarbeitung heute eine zentrale Rolle in der Wirtschaft innehat.

1.2 Einordnung und Umfang der Informatik

Versuchen wir die Wissenschaft Informatik in eine der klassischen Kategorien Natur-, Ingenieur- oder Geisteswissenschaft einzuordnen, so stossen wir auf Schwierigkeiten. Die Informatik versucht nicht, wie etwa die Physik oder die Chemie, Naturerscheinungen zu erklären. Alle Objekte, mit denen sich die Informatik beschäftigt, sind ingenieurmässig konstruiert, egal ob wir dabei an die Rechner (Hardware) oder deren Programme (Software) denken. Wir können die Informatik also nicht zu den Naturwissenschaften rechnen, obwohl sicherlich auch naturwissenschaftliche Kenntnisse in der Informatik benötigt werden und umgekehrt die Informatik in immer stärkerem Masse in die Naturwissenschaften eindringt. Eine Einordnung in die Ingenieurwissenschaften wird der Informatik auch nicht gänzlich gerecht. Informatik ist keine Hochfrequenz-, Nachrichten- oder Rechenmaschinentechnik. Ingenieurmässige Methoden bei der Konstruktion und bei der Erstellung verwertbarer Produkte (Hardware und Software) spielen jedoch eine grosse Rolle in der Informatik. Hierin liegt denn auch die Grenze zu den Geisteswissenschaften, die sich im wesentlichen auf Erkenntnisgewinn und Beschreibung von Sachverhalten beschränken. C.F. von Weizsäcker [Weizsäcker 71] ordnet deshalb die Informatik neben der Mathematik als Strukturwissenschaft [Hofstadter 80] ein.

Wir wollen nun durch Aufzählung einiger Schwerpunkte der Informatikforschung versuchen, den Umfang der Informatik zu umreissen und geben einige Stichworte zu aktuellen Strömungen:

- Softwaretechnik

  Methoden zur Beschreibung und Implementierung von Software. Sie umfasst alle praktischen und theoretischen Aspekte der Planung und Durchführung grösserer

Softwareprojekte. Ausserdem befasst sich die Softwaretechnik auch mit der Entwicklung von Benutzerschnittstellen, die die Programmentwicklung unterstützen sollen.

- Rechnerarchitektur

  Entwicklung von Rechnerstrukturen, Verarbeitungs- und und Kommunikationsschemata. Konzeption und Realisierung (z.B. durch Mikroprogrammierung) von Instruktionssätzen für machinennahe Programmierung.

- Automatentheorie und formale Sprachen

  Untersuchung von abstrakten Modellen für diskrete Prozesse (Automaten), Programmiertechniken und Systemen zur Beschreibung von Programmiersprachen.

- Komplexitätstheorie

  Mathematisch fundierte Untersuchung von Programmen (Algorithmentheorie) und Rechnersystemen (Statistische Modelle).

- Programmiersprachen und ihre Übersetzer

  Entwicklung benutzerorientierter Programmiersprachen. Implementierung dieser Sprachen auf Rechenanlagen aller Art. Dabei spielt vor allem die Weiterentwicklung von Portierungstechniken eine grosse Rolle. Entwicklung und Implementierung von Dialogsprachen zur Kommunikation mit Datenbanken und Informationssystemen.

- Betriebssysteme

  Programmsysteme zur Steuerung des Betriebsablaufes in Datenverarbeitungsanlagen mit dem Ziel eines optimalen Service für den Benutzer und einer optimalen Ausnutzung der eingesetzten Betriebsmittel.

- Systeme zur Informationsverwaltung

  Allgemeine Verfahren zur Speicherung und Bearbeitung strukturierter Datenbestände in Datenverarbeitungsanlagen, insbesondere Regeln zur Kennzeichnung und

Anordnung von Daten, darauf abgestimmte Algorithmen für Aufruf, Fortschreibung, Löschung und Ergänzung von Daten des Bestandes, Implementierung von benutzerorientierten Sprachen zur Programmierung entsprechender Bearbeitungsvorgänge.

- Rechnertechnologie

  Nutzung aktueller Möglichkeiten der Bauelemente-Technologie und Gerätetechnik für die Weiterentwicklung von DV-Anlagen (z.B. Miniaturisierung in Form vom Mikroprozessoren), Erforschung und Entwicklung neuer Technologien (VLSI) und Konstruktionen für die Bedürfnisse von DV-Systemen. Verfahren zur Automatisierung der Entwicklung und Fertigung (CAD/CAM).

- Verfahren zur digitalen Verarbeitung kontinuierlicher Signale

  Verarbeitung zeitlich veränderlicher Signale durch den Rechner, insbesondere im Zusammenhang mit der Mustererkennung (z.B. image processing) und der Überwachung und Steuerung physikalischer, chemischer, biologischer und technischer Prozesse. Entwicklung passender Verfahren und Einrichtungen zur Wandlung und Interpretation dieser Signale.

- Automatisierung technischer Prozesse

  Erforschung von Methoden zur Automatisierung technischer Prozesse im industriellen Produktionsbereich (Prozesskontrolle und Steuerung, Industrieroboter), im Transport- und Verkehrswesen, in der Energieverteilung, der Versuchsfeld- und Laborarbeit und der Haustechnik durch digitale Prozessrechnersysteme.

- Rechnergestütztes Planen, Entwerfen und Konstruieren

  Erforschung der Einsatzmöglichkeiten von DV-Systemen zur Rationalisierung von Planungs- und Entwurfsprozessen in komplexen technischen Systemen, wie sie etwa im Bauwesen, im Maschinenbau, in der Energie- und Nachrichtentechnik auftreten (z.B. CAD). Untersuchungen der Strukturen und Verfahrensweisen bestehender Bearbeitungsorganisationen sollen zu Ansätzen für die Entwicklung und Implementierung von

adäquaten rechnerunterstützten Bearbeitungsverfahren führen.

- Anwendung der Informatik in der Medizin

  Erforschung von Grundlagen und Methoden für Anwendungen der Informationsverarbeitung in der Medizin, die besondere gerätetechnische, programmiertechnische oder systemorganisatorische Anforderungen stellen.

- Anwendung der Informatik im pädagogischen Bereich

  Erforschung von Methoden und Verfahren der Informatik (einschliesslich Programmiersprachen) als Hilfsmittel des Lehrens, des Lernens, der Unterrichtsforschung und der Unterrichtsorganisation in der Ausbildung an Schulen, Hochschulen und anderen berufsbildenden Institutionen.

- Betriebswirtschaftliche Anwendung der Informatik

  Forschung auf dem Gebiet rechnerunterstützter Anwendungssyteme für betriebswirtschaftliche Aufgaben; Entwicklung von ökonomischen Methoden zur Unterstützung der Einsatzvorbereitung und des Betriebes von Datenverarbeitungssystemen.

- Anwendung der Informatik in Recht und öffentlicher Verwaltung

  Forschung auf den Gebieten der rechnerunterstützten Dokumentations- und Informationssysteme im Rechts- und Staatswesen, der rechnerunterstützten Rechtssetzung, Rechtsanwendung und Rechtskontrolle und der Verwaltungsautomatisierung.

Allen diesen Teilgebieten ist eine bestimmte Methodik der Problemlösung gemeinsam, die man "Algorithmik" nennt.

1.3 Generelle Literatur zur Informatik

An dieser Stelle wollen wir einige Bücher empfehlen, die uns geeignet erscheinen, den hier gebotenen Stoff zu vertiefen. Wir unterscheiden dabei zwischen Literatur zur Informatik allgemein, anderen empfehlenswerten Lehr-

büchern der Algorithmik und Büchern zur Vertiefung des in diesem Buche behandelten Stoffes.

Geschichte der Informatik:

- The Origins of Digital Computers [Randell 73],
- A History of Computing in the Twentieth Century [Metropolis 79]
- History of Programming Languages [Wexelblatt 81]

Einführende Lehrbücher:

- Systematisches Programmieren [Wirth 72],
- Algorithmen und Datenstrukturen [Wirth 75],
- Informatik I und II [Bauer 71]

Ergänzende Lehrbücher:

- The Science of Programming [Gries 81],
- Algorithmische Sprache und Programmentwicklung [Bauer 81],
- Fundamental Structures of Computer Science [Wulf 81],
- The Art of Computer Programming [Knuth]
  Umfassendste Darstellung der Algorithmik in mehreren Bänden; auch als Nachschlagewerk geeignet.
- Fundamentals of Data Structures [Horowitz 76]
- Data Structures and Algorithms [Aho 82]
- Structured Programming [Dahl 72]

2. Algorithmen

Thema dieses Buches sind die Algorithmen. In diesem Kapitel wird versucht, diesen grundlegenden Begriff der Informatik anhand von Beispielen aus dem täglichen Leben, in denen Algorithmen für Menschen beschrieben sind zu verdeutlichen. Es wird gezeigt, dass zum Erkennen und Beschreiben von wesentlichen Merkmalen eines Algorithmus Abstraktionsvermögen notwendig ist. Das Kapitel endet mit einer intuitiven Beschreibung und der Angabe wesentlicher Eigenschaften von Algorithmen.

2.1 Algorithmen im Alltag

In nahezu allen Lebensbereichen, in Produktion, in Verwaltung, Freizeit und - nicht zu vergessen - in der Wissenschaft, also sozusagen im täglichen Leben, begegnen wir Anleitungen, wie man etwas macht, Rezepten, nach denen man etwas zubereitet, und Vorschriften, nach denen man sich richtet (oder vielleicht auch nicht).

Der gemeinsame Begriff für solche Anleitungen, Rezepte und Vorschriften heisst Algorithmus. Die Bedeutung dieses Begriffs wollen wir zunächst intuitiv herausarbeiten.

Als Vorbereitung auf die Frage, wie man einen Rechner programmiert, analysieren wir einige solcher Anleitungen für Menschen.

2.1.1 Benutzen eines Münzfernsprechers

Ein erstes Beispiel eines Alltagsalgorithmus ist die Anleitung zur Benutzung eines Münzfernsprechers:

1. Handapparat abnehmen
2. Wählton und Speicherbeleuchtung abwarten
3. Mindestens 2x10 Pfg. einwerfen
4. Wählen

Beim Durchlesen dieses Algorithmus fallen einige wesentliche Punkte auf:

a) Diese Anleitung ist ausführbar. Hingegen ist z.B. die Feststellung "Münzfernsprecher auf Strassen und Plätzen sind im allgemeinen ununterbrochen betriebsbereit" nicht ausführbar und damit kein Algorithmus.

b) Die Ausführung des Algorithmus erfolgt schrittweise. Die Folge von Schritten bei der Ausführung eines Algorithmus nennen wir einen von diesem Algorithmus beschriebenen Prozess. Den Ausführenden des Algorithmus, der in diesem Falle ein Mensch ist, nennen wir den Prozessor.

c) Jeder Schritt in der Ausführung besteht selbst wieder aus der Ausführung eines oder mehrerer (anderer) Algorithmen, die in der Anleitung durch einen Namen (wie: wählen) angedeutet werden. Diese benannten Algorithmen werden in der Anleitung als elementar aufgefasst. Der gegebene Algorithmus ist zusammengesetzt aus elementaren Algorithmen. Wer diesen Algorithmus ausführen will, muss eine Menge wissen: er muss die Sprache, in der der Algorithmus beschrieben ist, d.h. Deutsch, kennen, er muss wissen, was ein Wählton ist und wie man 10 Pfennigstücke einwirft. Anders gesagt: Er muss die elementaren Algorithmen kennen und in der Lage sein, sie auszuführen.

d) Von einem Algorithmus fordern wir, dass er hinreichend genau ist, d.h., dass jeder der Schritte und die intendierte Reihenfolge ihrer Ausführung unmissverständlich sind. Die Formulierung eines Algorithmus muss aber auch auf einer wohlgewählten Ebene des Details aufhören (die Ebene der elementaren Algorithmen), weil man sonst in einem Sumpf von immer detaillierteren Beschreibungen versinkt: Um das Abnehmen des Handapparates zu erklären, könnte man angeben, welche Hand zum Handapparat zu führen ist (unterschiedlich für Links- und Rechtshänder), welche Kraft zum Abnehmen des Handapparates benötigt wird (natürlich vektoriell: Betrag der Kraft in Newton und Richtung), und, dass der Teil des Handapparates, an dem die Zuleitung nicht angebracht ist, mit der flachen Seite an das Ohr (rechts oder links, unterschiedlich für Links- und Rechtshänder) zu führen ist.

Wir könnten weiter definieren, welche Muskeln beim Ausstrecken der Hand betätigt werden müssen und finden bei unserer Beschreibung kein Ende. Die Genauigkeit der Beschreibung eines Algorithmus enthält immer die Wahl einer der Komplexität des Algorithmus angemessenen und für den Ausführenden verständlichen und geeigneten Ebene des Details und bezieht sich nur auf diese Ebene.

e) Ein Algorithmus ist eine endliche Beschreibung eines Prozesses. Es ist denkbar, dass durch eine endliche Beschreibung ein Prozess definiert wird, der nicht nach endlicher Zeit beendet wird (nicht terminiert). Bei den Beispielen in diesem Buch und bei den Übungen beschäftigen wir uns ausschliesslich mit Algorithmen, die nach endlicher Zeit terminieren (sollen). Es gibt aber durchaus auch Algorithmen mit praktischem Nutzen, die nicht terminieren, zum Beispiel in der Prozessdatenverarbeitung und in Betriebssystemen.

2.1.2 Kochrezept

Die Zubereitung von Mayonnaise erfordert einiges Geschick. Ein Hobbykoch wird deshalb im Kochbuch nachsehen, wie er Mayonnaise herstellen kann. Er findet dort folgendes Rezept:

> Mayonnaise
>
> 2 Eigelb, Salz, Pfeffer, 1/8 Liter Salatöl, ein Teelöffel Senf, ein Teelöffel Zitronensaft.
>
> Wenn die Zutaten aus dem Kühlschrank kommen, müssen sie zuerst auf Zimmertemperatur erwärmt werden, damit die Mayonnaise beim Rühren nicht gerinnt.
>
> Eigelb, Salz, Pfeffer, Senf und Zitronensaft solange kräftig mit einem Schneebesen schlagen, bis die Masse homogen ist. Danach tropfenweise, unter ständigem Rühren, die Hälfte des Öls hinzufügen. Der Rest des Öls kann dann rasch unter die steife Masse gerührt werden.

Für einen halbwegs erfahrenen Koch ist dieses Rezept eine genaue, ausführbare, endliche Beschreibung eines endlichen Prozesses, die er unter dem Namen "Zubereitung von Mayonnaise" kennt.

Auch bei diesem Algorithmus fallen einige wichtige Merkmale auf:

a) Einige Teile des Algorithmus sollen nacheinander (sequentiell, seriell), andere Teile gleichzeitig (parallel) ausgeführt werden. Teilweise können sie auch in beliebiger Reihenfolge oder sogar gleichzeitig ausgeführt werden (kollateral).

   Sequentiell:
   Erst wird das Eigelb mit Salz, Pfeffer, Senf und Zitronensaft zu einer homogenen Masse geschlagen und dann das Öl zugegeben.

   Parallel:
   Das Öl wird "unter ständigem Rühren" zugegeben. Wenn die Zugabe von Öl und das Rühren nicht gleichzeitig ausgeführt werden, gelingt die Ausführung des Algorithmus nicht - die Mayonnaise gerinnt.

   Kollateral:
   Die Vermengung von Eigelb, Salz, Pfeffer, Senf und Zitronensaft kann in beliebiger Reihenfolge ausgeführt werden. Wenn genügend Prozessoren (z.B. Hände) zur Verfügung stehen, kann die Ausführung sogar parallel vorgenommen werden. Man kann aber auch die Vermengung in fünf sequentiellen Schritten durchführen. Das Resultat des Algorithmus ändert sich hierdurch nicht. Lediglich seine Abarbeitung wird etwas länger dauern.

b) Wir finden in diesem Kochrezept zwei wichtige Mechanismen zur Zusammensetzung von Algorithmen: die Wiederholung und die Auswahl. Wir fügen solange Öl unter ständigem Rühren hinzu, bis die Hälfte des Öls verbraucht ist. Die zu wiederholende Handlung (Tropfen hinzufügen unter ständigem Rühren) wird beendet, sobald eine bestimmte Bedingung (die Hälfte des Öls ist verbraucht) erfüllt ist. Eine Art der Auswahl finden wir im ersten Satz des Kochrezepts: Abhängig

von einer Bedingung (Zutaten kommen aus dem Kühlschrank) wird ausgewählt zwischen zwei Handlungen. Wenn die Bedingung erfüllt ist, werden die Zutaten auf Zimmertemperatur erwärmt, sonst brauchen wir nichts zu tun. Die zweite Alternative (nichts tun) wird nicht explizit im Rezept erwähnt. Die zusammengesetzten Algorithmen beschreiben also eine ganz spezielle Ausführungsreihenfolge für ihre Komponenten.

c) Am Anfang werden die benötigten Zutaten benannt und bereitgestellt, anschliessend werden sie verarbeitet. Ein Algorithmus beschreibt Operationen auf benannten Objekten wie z.B. das Hinzufügen von Öl. Die Objekte werden teilweise zur Ausführung des Algorithmus benötigt, wie die Zutaten Eigelb, Salz, Pfeffer, Öl, Senf, Zitronensaft (Eingabe). Teilweise entstehen sie auch erst durch die Ausführung des Algorithmus, wie hier die Mayonnaise (Ausgabe), und wieder andere existieren alleine während des Prozesses (lokale Objekte).

d) Objekte können elementar sein, d.h. im Algorithmus mit einem Namen angedeutet und als unteilbare Einheiten aufgefasst werden, oder aus anderen Objekten zusammengesetzt sein. Elementar sind in diesem Beispiel etwa das Salz oder der Pfeffer. Ein zusammengesetztes Objekt wäre der Münzfernsprecher aus dem vorigen Beispiel, der aus einem Handapparat, einer Wählscheibe usw. besteht. Auch bei Objekten bezieht sich die Bezeichnung 'elementar' auf eine bestimmte Ebene des Details. Für einen Chemiker ist Salz kein elementares Objekt. Der Atomphysiker interesiert sich sogar für die Zusammensetzung der im Salz enthaltenen Natrium-Atome.

e) Es gibt einen starken Zusammenhang zwischen den Algorithmen und den Objekten: Die Wahl der elementaren Objekte bestimmt die Ebene des Details der darauf arbeitenden Algorithmen, und umgekehrt legt die Wahl der elementaren Algorithmen die Art der Objekte fest, die sie manipulieren.