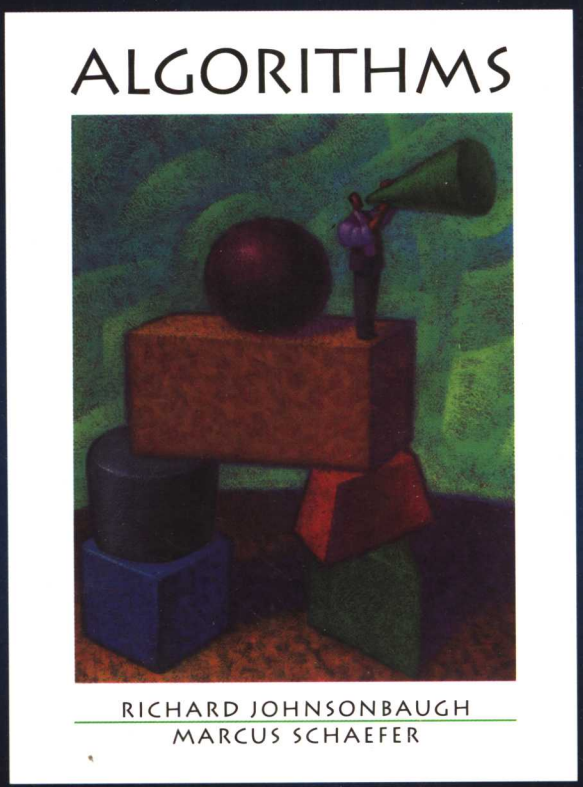
PEARSON Prentice Hall

国外计算机科学技术经典教材

# 大学算法教程

RICHARD JOHNSONBAUGH　MARCUS SCHAEFER　著




方存正　曹旻华　明　译
吴洪来　审校

PEARSON Education

清华大学出版社

# 大学算法教程

[美] Richard Johnsonbaugh, Marcus Schaefer 著

方存正 曹旻华明 译

吴洪来 审校

清华大学出版社

北京

## 内 容 提 要

本书是美国德保罗大学（DePaul University）教授 R.Johnsonbaugh 等人长期从事算法课程教学经验的结晶，是一本关于算法基础知识和基本方法的教科书。内容包括：算法必备的数学基础、数据结构和描述算法的语言与记号；常用算法的设计分析及其正确性证明；NP 和 NP 完全问题的特征及其近似处理方法。

全书含 300 多个生动有趣的算法实际示例和 1450 多道习题，从经典方法到最新成果，层层剖析、逐步深入。根据相关方法的重要程度，详简适度地作了富有启发性的介绍和论证。

本书可以作为大学计算机科学技术及相关专业本科生和研究生算法课程的教材，也可作为高职相关专业教学的参考用书。



**图书在版编目（CIP）数据**

大学算法教程/（美）约翰森堡（Johnsonbaugh,R.），（美）谢菲尔（shaefer,M.）著；方存正，曹旻，华明译. —北京：清华大学出版社，2007
书名原文：Algorithms

ISBN 978-7-302-15040-4

I. 大… II. ①约… ②谢… ③方… ④曹… ⑤华… III.算法—教材
IV. 0242.23

中国版本图书馆 CIP 数据核字（2007）第 051777 号

**责任编辑**：陈 洁
**责任校对**：刘雪莲
**责任印制**：科 海

**出版发行**：清华大学出版社
http://www.tup.com.cn
c-service@tup.tsinghua.edu.cn
**社 总 机**：010-62770175
**投稿咨询**：010-62772015
**地 址**：北京清华大学学研大厦 A 座
**邮 编**：100084
**邮购热线**：010-62786544
**客户服务**：010-62776969

**印 装 者**：北京市鑫山源印刷有限公司
**经 销**：全国新华书店
**开 本**：185×260 **印 张**：39
**字 数**：949 千字
**版 次**：2007 年 6 月第 1 版
**印 次**：2007 年 6 月第 1 次印刷
**印 数**：1～4 000
**定 价**：69.80 元

# 译者序

算法是涉及“计算”的众多学科的一个共同话题。它包括算法的设计、分析、实现和应用。算法的基本问题是：什么样的问题能有效地用计算机自动地加以解决；什么样的问题则不能。自20世纪30年代以来，这个原本由数学家关心、研究的课题，已经发展成为计算机科学的有力武器。算法已是今天计算机科学技术专业及相关专业的重要基础课程。实际上，它对于这些专业领域的从业人员来说，往往是终身要与之打交道的一门学问。因为任何一个计算机应用系统，都要经历“问题的提出——数学模型的建立——算法的设计与评测——程序代码的编制——计算机系统上的实现”几个阶段，这整个过程中，算法处于承上启下的关键地位。但是在我国大学相关专业的课程设置中，即使是计算机专业，算法的系统学习和训练也是被安排为本科高年级或研究生课程。实际上，算法的理念应当贯穿于专业教学的全过程。

要把算法教科书编写得有血有肉、生动有趣（更不用说是出神入化），确实需要相当的功底。本书作者R. Johnsonbaugh等人长期讲授算法课程，有着丰富的教学经验积累。实践证明，使用恰当的实例，将会起到穿针引线的作用。本书就是选用了大量的发生在生活、学习中的应用实例，理论结合实际地进行阐述。书中除了少数章节和习题需要概率论和数学分析知识外，尽量使用初等方法，由浅入深、由此及彼地进行分析，努力做到循循善诱，全书不乏引人之处。300多个经过验证的实例有助于学生抓住各种算法的精神实质，领会应用，并逐步学会设计和评价算法。

本教材主要由以下几个部分组成。第1部分包括第1~3章，是有关算法的一些基础知识的复习和回顾。其中包括算法的伪代码描述介绍、算法的表示、正确性证明以及算法分析中常用的数学方法及各种常用记号的表示。这部分还讲述了本书用到的基本数据结构，包括图、树、链接表、队列和堆等。

第2部分包括第4~9章，是一些常用算法和应用，包括搜索、排序和选择，以及分治、贪心、动态规划等基本方法。这部分的内容相当丰富，除常见的广度优先搜索、深度优先搜索、拓扑排序、归并排序、快速排序、计数排序、基数排序外，还包括文本搜索、近似模式匹配、正则表达式匹配和回溯。对分治法的介绍是结合平铺问题、最近点对寻找问题、矩阵乘法问题的实例进行的。贪心算法是结合硬币兑换、Huffman编码和连续背包问题进行讨论的。动态规划方法的介绍从计算Fibonacci数列开始，并针对硬币兑换问题将动态规划算法和贪心算法作了比较，然后进入矩阵乘法、最长公共子串问题的动态规划方法的讨论，并在有关部分介绍了许多行之有效的算法名著或结果。

第3部分包括第10~11章，讨论了P与NP问题，以及NP完全性问题的处理方法。这是本书内容中相当精彩的部分，包括不确定算法和NP；NP完全性问题和归约；各种各样的NP完全性问题，例如独立集和团、图的着色、移动电话网络、背包问题、调度问题、扫雷游

戏、纵横字谜、装箱问题、密码问题、蛋白质褶和计算生物学等等，以及它们之间的等价性。所用的方法包括蛮干法、随机方法、近似方法、参数化和启发式试探方法等。

第4部分包括第12章，介绍了并行和分布算法。从求最大值的并行化方法着手，介绍并行随机存取机器（PRAM）时，涉及了半群算法和加速级联、并行前缀、指针跳转等方法；介绍排序网络时，引入了双调排序、归并和排序、排序网络的0-1原理等；在介绍并行体系结构时，讨论了并行体系多处理器之间的拓扑结构及其标记问题等；在介绍分布式算法时，讨论了网上广播和主播机的选择等问题。

在上述各个部分，除了阐明众所周知的许多经典算法外，还循序渐进地介绍了相关算法的改进和最新成果。每章还附有一个“备考”栏目，介绍有关专题的历史沿革和最新进展，供读者进一步查阅，以扩充加深自己的知识和技能。

全书共有1100多道节后习题，有助于学生理解相关内容，检查自己对知识的掌握程度。各章末都有本章习题，共有350多道。从整体上说，它们比节后习题要稍难些。其中有些题目相当难，甚至可以当作一个小的课题，可能需要教师的适当指导才能完成。

本书由方存正、曹旻、华明三位老师翻译，方存正负责序言、第1章、第5~7章，华明负责第2~4章，曹旻负责第8~12章，习题选解由各章译者分别完成。方存正老师还对书的后半部分的译稿作了订正。吴洪来老师负责全书统稿和审校。虽然译者尽了努力，但仍可能存在不妥和谬误之处，殷切期望读者和老师们不吝赐教。

2007年2月

# 前　言

## 本书写作的缘由

本书是为本科高年级学生或研究生的算法课程编写的，它基于我们教授这门课程25年的经验。本书的主要特点是：

- 强调设计技术。
- 展示算法多么有趣并令人神往。
- 包含实际的应用。
- 提供大量验证过的实例和练习题。

设计者在面对一个新的计算问题时，往往就可以使用本书中的某个算法来解决，顶多作少许的改动或调整。但是，对有些问题，书中的任何一个算法都会无能为力。因这个缘故，我们在书中介绍了一整套的设计技术，可以利用这些技术来解决问题；并且它们能够帮助读者进一步提高洞察力，以辨别哪些技术更有希望取得成功。关于论述NP完全问题及其解决办法的那几章，还讲述了怎样识别那些难以解决的问题，以及在这种情况下可以采用哪些技术。

关于算法的学习和工作是很有趣和令人神往的。算法的设计乃是一项富有想象力的创造性的任务，它要对看上去云遮雾障的种种新问题和老问题提出解决办法。我们相信，为了成功，一定要体会一个新问题提出的挑战所带来的乐趣。为了这个目的，本书比同类书籍中包括了更多综合性和消遣性的例子和练习题。对付一个未解的问题，常会被人认为是面对一个威胁，而不是当作一种施展才智的良机。我们希望这些例子和习题能够有助你消除这种威胁感。

本书中，算法实际应用的例子有7.5节的数据压缩，还有9.4节中的用于作为UNIX的agrep命令部分实现的Boyer-Moore-Horspool算法。书中大部分节的第一段都会先介绍一个激发学习兴趣的例子。5.3节的最近点对问题由一个模式识别例子开始，8.4节涉及最长公共子串问题，从蛋白质分析的讨论开始。

经验是学习、研究算法设计和分析的最佳途径。为此，我们提供了大量的经过验证的例子和习题。这些经验证的例子展示如何来应对算法，而习题让读者反复训练这些技术。全书有300多个经验证的例子。这些例子阐明了如何开发算法，展示了理论的应用，提供了证明，有助于激励学生对相关知识的追求。本书包含1450多道习题，有易有难，全都经过课堂教学的检验。习题的表述力求清晰、精确。对于学生来说，及时的反馈十分重要，所以每节后的习题共有三分之一（题号后加“S”标记）在书后附有答案。其余节后习题的答案只供教师参考（见后面“教师补遗”一节）。

## 先修课程

关于计算机科学的主要先修课程是数据结构，应包括堆栈、队列、链接表、树和图。主要的数学先修课程是离散数学，应覆盖逻辑、渐近标记（例如“大O”标记）和递推关系及其迭代解法。我们不采用诸如生成函数这样的高等方法，但在某一二处会用到微积分的一些基本概念。第2~3章概述了本书所用到的数学知识和数据结构知识，可以根据需要，将这部分内容作为参考资料、复习提纲，或并入算法课程进行讲授。

## 本书内容

在本书开头3章（包括引言、数学知识和数据结构）之后，紧接着的5章侧重于算法设计技术。

第4章主要讲述搜索技术，包括一些较新颖的应用，如数字图像中的区域查找。

第5章介绍分治技术。涉及的问题有平铺问题、平面上最近点对的查找、Strassen矩阵乘法算法。

第6章是关于排序和选择，其中的许多算法都使用分治法开发。

第7章讲述如何使用贪心方法开发算法。先讲贪心方法在简单场景（硬币找零）的使用，再讲Kruskal算法、Prim算法、Dijkstra算法、Huffman算法，以及连续背包问题的解。

第8章是关于动态程序设计的技术。像第7章一样，先讲动态程序设计如何用在简单场景（计算斐波纳契数列），再重新回到硬币找零问题（第7章贪心方法），并将动态程序设计和贪心方法两者进行比较。然后讨论矩阵最佳分组、最长公共子串问题，以及Floyd算法和Warshall算法。

第9章讨论文本搜索技术，包括Knuth-Morris-Pratt算法和Boyer-Moore-Horspool算法，以及不精确搜索法。

第10章研讨NP完全性问题——这是识辨和了解算法局限性的一个理论方法。我们将从不同的领域，例如移动电话网络、游戏和生物计算，举很多例子来说明NP完全性无处不在的普遍性。

人们普遍认为，NP完全性问题不能被算法有效解决。但是不管怎样，这些问题确实会在实际应用中产生，必须在实践中得到解决。第11章我们将针对NP完全性提出一系列来自实际和理论的技术，力图在不同程度上求解NP完全性问题。讨论的方法包括近似法、参数化法，以及试探法。

第12章介绍用于并行体系结构的一些基本算法，包括用于PRAM的算法和排序网络的算法，还要介绍分布环境的计算。

## 教学提示

书中的每一节（除了引言性的12.1节）节后都有习题。全书共1100多道节后习题，其中有些检查对书中材料的基本理解（例如，要求回答一个算法的描述），有些则检查对材料深入掌握的程度（例如，研讨选用其他算法）。比较难的习题都标有星号*。

每章的结尾，在本章习题之前，有一个备考栏目，其内容是对进一步阅读的建议和参考资料。本章习题整合了该章的材料，有的附有提示。全书包含的各章练习题共有350多道。从总体上说，它们要比节后习题难。标有两颗星记号的则是一些相当难的题目，也许需要教师的指导，有些甚至可以作为一个小课题。

对于问题下界的研究，都在有关问题的章节里进行讨论，而没有单立成一章。例如，在介绍一些排序算法后，会讨论基于比较的排序的下界（6.3节）。

我们将讲述和讨论许多新近的成果，例如，参数化表示的复杂性，这是当前的一个研究领域。

算法用伪代码写就，与C、C++和Java类语言的语法接近。我们之所以没有采用这些语言所用的数据类型、分号和一些冷僻的特性，是因为用实际代码来书写算法，反而会表达不清，对于不熟悉该种语言的人来说就更难读懂。本书所用的伪代码在1.2节中有完整的描述。

书中用插图对概念进行图解，形象地说明算法如何运作，解释证明，让书中的材料更生动有趣。有些图用以说明定理的证明，这些插图说明提供了进一步的解释，有助于更透彻地理解证明。

本书特别注意寻求对正确性最直接和最易理解的证明[作为例子，见定理7.2.5，Kruskal算法和Prim算法的正确性的导出，还有Dijkstra算法的正确性证明（定理7.4.5）]。

我们会举出一些例子和论据来说明，对算法的时间界限很严峻。例如，见7.3节中的“下界时间估算” 小节，它表明使用二叉堆的Prim算法的最坏情况时间上界相当严峻。这后面是定理7.5.4，它表明Huffman算法的最坏情况时间上界相当严峻。

## 教师补遗

英文版的“教师指导书”中有全部节后习题的答案，但是它没有包括在本书中。采用或试用本书的教师可以免费从Prentice Hall代理商处获取。

## 网站

www　网站http://condor.depaul.edu/~rjohnson含有：

- 对一些特定论题的进一步阐述和关于算法的更多信息。标有图标 *www* 的小节，表示该节有关的信息可以在此网站上或链接到另一网站得到。
- 算法动画软件。
- PowerPoint演示材料。
- 有关本书中一些论题的最新消息和参考材料。
- 补充材料。
- 计算机程序。
- 勘误表。

# 致谢

本书编写过程中得到很多人的帮助。感谢以下各位，他们评阅了全部或部分手稿：Ashland大学的Iyad A. Ajwa，Abilene Christian大学的Michael D. Frazier，California大学Irvine分校的Norman Jacobson，Rochester大学的Joel Seiferas，Depaul大学的Amber Settle，以及在Chattanooga的Tennessee大学的Jack Thompson。

DePaul大学的Ljubomir Perkovic对本书的大部分材料都经过了课堂检验，仔细阅读了全部材料，并提出了详尽而宝贵的意见。

感谢Massimo DiPierro，他为本书网站提供的算法动画软件作出了贡献。

感谢文字编辑Patricia Johnsonbaugh，他查校了大量细节，捉出了文中无意的错误，提出了不少改进的建议。

感谢DePaul大学计算机科学、电信和信息系统学院院长Helmut Epp，他为我们编写这本书给予了时间上的支持和鼓励。

感谢给予我们一贯支持的Prentice Hall的工作人员，他们是副社长、工程和计算机科学部主编Marcia Horton，计算机科学部执行采集编辑Petra Recter，计算机科学部资深采集编辑Kate Hargett，计算机科学部总编辑Camille Trentacoste，计算机科学部制作编辑Kathy Kasturas、助理编辑Sarah Parker，计算机科学部执行营销经理Pamela Shaffer，以及计算机科学部营销助理Barrie Reinhold。

R.J.

M.S.

# 目　录


**第1章　引言** ..... 1

1.1　算法 ..... 1

1.2　表述算法的伪代码 ..... 4

1.3　现状 ..... 8

1.4　未来发展 ..... 10

备考 ..... 12

本章习题 ..... 12

**第2章　算法涉及的基本数学概念** ..... 14

2.1　定义、记号和基本结论 ..... 14

2.2　数学归纳法 ..... 26

2.3　算法分析 ..... 34

2.4　递推关系 ..... 46

2.5　图 ..... 57

2.6　树 ..... 73

备考 ..... 79

本章习题 ..... 79

**第3章　数据结构** ..... 83

3.1　抽象数据类型 ..... 83

3.2　堆栈和队列 ..... 85

3.3　链接表 ..... 94

3.4　二叉树 ..... 101

3.5　优先队列，二分堆阵，堆阵排序 ..... 112

3.6　不相交集 ..... 126

备考 ..... 136

本章习题 ..... 137

**第4章　搜索** ..... 139

4.1　对分搜索 ..... 139

4.2　深度优先搜索 ..... 145

4.3　广度优先搜索 ..... 152

4.4　拓扑排序 ..... 158

4.5　回溯法 ..... 163

备考 ..... 175
本章习题 ..... 176

**第5章 分而治之 ..... 179**

5.1 平铺问题 ..... 179
5.2 归并排序 ..... 184
5.3 寻找最近点对 ..... 189
5.4 Strassen的矩阵乘法算法 ..... 195
备考 ..... 198
本章习题 ..... 198

**第6章 排序和选择 ..... 201**

6.1 插入排序 ..... 201
6.2 快速排序 ..... 204
6.3 排序问题的下界 ..... 213
6.4 计数排序和基数排序 ..... 215
6.5 选择 ..... 220
备考 ..... 224
本章习题 ..... 224

**第7章 贪心算法 ..... 227**

7.1 硬币兑换 ..... 227
7.2 Kruskal算法 ..... 230
7.3 Prim算法 ..... 237
7.4 Dijkstra算法 ..... 247
7.5 霍夫曼编码 ..... 252
7.6 连续背包问题 ..... 262
备考 ..... 267
本章习题 ..... 267

**第8章 动态规划算法 ..... 269**

8.1 计算斐波那契数列 ..... 269
8.2 硬币兑换问题再探讨 ..... 273
8.3 矩阵乘法 ..... 280
8.4 最长公共子串问题 ..... 284
8.5 Floyd算法和Warshall算法 ..... 290
备考 ..... 300
本章习题 ..... 300

**第9章 文本搜索 ..... 304**

9.1 简单的文本搜索 ..... 305

9.2 Rabin-Karp算法 307
9.3 Knuth-Morris-Pratt算法 314
9.4 Boyer-Moore-Horspool算法 325
9.5 近似模式匹配 330
9.6 正则表达式匹配 339
备考 351
本章习题 351

**第10章 P和NP问题 356**
10.1 多项式时间 356
10.2 不确定算法和NP 362
10.3 可归约性和NP完全性 375
10.4 NP完全性问题 386
10.5 NP完全性进一步探讨 393
备考 398
本章习题 399

**第11章 NP完全性问题的处理 408**
11.1 蛮干法 410
11.2 随机方法 417
11.3 近似方法 423
11.4 参数化方法 436
11.5 启发式搜索 448
备考 456
本章习题 457

**第12章 并行和分布算法 463**
12.1 引言 463
12.2 并行随机存取机器（PRAM） 467
12.3 排序网络 487
12.4 并行体系结构 501
12.5 分布式算法 519
备考 529
本章习题 530

**参考文献 534**

**习题选解 541**

# 第1章
# 引言

简略地说，一个**算法**（algorithm）就是一步一步地解决某种问题的方法。其实，问题求解的这样一种方法并非什么新的发明，“algorithm”这个词源自9世纪波斯数学家al-Khowārizmī的名字。当今，“算法”通常是指能够被计算机执行的一种解法。在本书中，我们所关注的主要是能够在“传统的”计算机上执行的算法，也即指能在一条一条地执行指令的单处理器的计算机上，诸如个人计算机这样的计算机上执行的算法。

1.1节将详细讨论“算法”的概念。1.2节讲述伪代码，这是一种说明算法的精确办法。1.3节讨论算法技术的现状。1.4节简要介绍一下已经提出的实施计算的一些其他途径，这些方法将会对未来的算法和计算产生深远的影响。

## 1.1 算法

算法一般具有以下特性：

- **输入**。算法所接受的输入。
- **输出**。算法产生的输出。
- **精确性**。每一步都被精确地加以说明。
- **确定性**。每一步执行的中间结果都是惟一的，只取决于输入和前面步骤的结果。
- **有穷性**。算法会终止；就是说，经过许多指令有限次数的执行后会停止。
- **正确性**。算法产生的输出是正确的。
- **通用性**。该算法可以用于一个输入的集合。

**例1.1.1　在三个数中寻找最大的一个**。考虑如下算法，从三个数$a$、$b$和$c$中寻找最大值：

1. $x = a$。
2. 若$b > x$，则$x = b$。
3. 若$c > x$，则$x = c$。

运算符 = 是**赋值运算符**；$y = z$的意思就是将$z$的值复制到$y$，而$z$保持不变。这个算法的思路就是将这几个数一个个地查看，把看到的那个最大的值复制到变量$x$。最后，$x$就等于这三个数中的最大数。

我们来验证这个算法具备了上面列出的那些特性。它接受$a$、$b$、$c$三个数值作为输入，

回送数值$x$作为输出。

这一算法的每个步骤都表述得足够精确，所以它完全能够用一种程序设计语言来编写，并被计算机执行。

给定输入的值，这个算法的每个中间步骤都产生一个惟一的结果。例如，如果给定的输入为：

$$a=1，b=5，c=3$$

那么，在算法的第2行，不管谁来执行，$x$一定会置成5。

经过有限多步（三步）后这个算法就终止，对问题（找出三个输入值中的最大值）给出正确的答案。

这个算法是通用的；它能够找出任意三个数中的最大值。 □

算法的研究有两个主要任务：设计解决一个特定问题的算法，以及对一个给定的算法进行分析。这两个任务是紧密相关的。分析算法能深入地理解算法，从而有助于设计出新的算法。

**算法设计**（algorithm design）与其说是一门科学，不如说是一门艺术。最说明问题的就是唐纳德·克努斯（Donald Knuth，见Knuth，1997，1998a，1998b）所著的有关这个论题的那本极其有名的精湛教科书《计算机程序设计》（The Art of Computer Programming）。不过，就像其他艺术（或者科学）一样，也有一些基本的算法设计技术能够被认定。我们将会考察一些重要的算法设计技术：搜索技术（第4章）、分治法（第5章）、贪心算法（第7章）和动态规划（第8章）。第6章讨论排序和选择，并且使用了第5章介绍的许多分治技术。

在**算法分析**（analysis of algorithms）中，我们要求回答如下一些问题：

- **正确性**。对一个问题给出的算法能否解决这个问题？
- **终止性**。经过有限个步骤后，该算法是否一定会停止？
- **时间分析**。该算法要执行多少条指令？
- **空间分析**。执行该算法需要占用多大的存储空间？

贯穿本书，问题结果的正确性一直都会被加以关注。通常会对一个给定算法的正确性构造一个数学证明。一个算法是否会终止，往往容易看得清（如例1.1.1），或者遵循算法的时间分析得到的结果。

然而，单单有正确性和终止性是不够的。能够正确解决问题的一个算法，也许因为在计算机上所需执行的时间太长，或者所占用的存储空间太大，实际上还是不能用。

www **例1.1.2 密码术。密码术**（cryptography）是指人们交换信息时不让非授权用户擅自截获的方法。从正确性和终止性的观点看，目前所采用的大多数密码系统都不难破译。假定我们知道加密函数$e(s, k)$，它有明码电文$s$和密钥$k$，产生加密的电文$e(s, k)$。一般说来这个加密算法都可以被掌握，并且迅速地加以计算（为了加密必须执行这个算法）。这里就有一个算法，它可以根据一个给定的加密电文$c$，产生相应的明码电文$s$（如果有相应明码电文的话）：对所有可能的密钥$k$和所有可能的明码电文$s$，计算$e(s, k)$，并把结果与$c$比较。如果找到一个$s$和$k$，使得$e(s, k)=c$，则回送$s$和$k$。这种蛮干的办法可以破译很多现有的加密

系统；但是，这种办法所花费的时间实在太长，并没有实用价值。 □

知道或者能够估算一个算法所需的时间和空间是非常重要的。知道了算法所需的时间和空间，就可以对解决同一问题的不同算法进行比较。例如，如果解决某个问题，一个算法要$n$步，而另一个算法要$n^2$步，那么只要所需的空间合理，我们一定会喜欢第一个算法。在第2章我们将给出技术定义，对算法所需的时间和空间作严格的表述。

第9章讨论用于文本搜索的几个专门算法；第10章介绍一类还未找到时间上有效的算法的问题，而许多这样的问题却相当实用（例如，有些调度问题就属于这类问题），因而必须解决它们。第11章将要讨论对付这些难题的技术。

最后一章（第12章）讨论并行和分布算法。并行和分布算法要求多处理器，而不是像个人计算机那样的单处理器，这样，多条指令就能够并行（即同时）执行。通过同时执行几条指令，就能达到更快解决问题的目的。

## 习题

1S[1]. 写一个算法，在$a$、$b$、$c$中找出最小的那个。

2. 写一个算法，在三个不同的值$a$、$b$、$c$中找出中间的那个。

3. 将在小学里学到的两个十进制正整数相加的标准方法写成一个算法。

4S. 在下面表述的算法中，是否缺失了算法某些应有的特性——输入、输出、精确性、确定性、有穷性、正确性、通用性？请说明道理。其中输入是一个整数集$S$和一个整数$m$。输出是和为$m$的$S$子集的全体。

1）列出全部$S$子集以及它们的和。

2）一个个查看第1步列出的子集，把每个和等于$m$的子集输出。

5. 查阅使用手册，对一个磁带录像机进行编程。具备算法的哪些特性——输入、输出、精确性、确定性、有穷性、正确性、通用性？缺失算法的哪些特性？

6. 哥德巴赫猜想指出，每个大于2的偶数都是两个质数之和。这里有一个算法，验证哥德巴赫猜想是否正确：

1）令$n=4$。

2）若$n$不是两个质数之和，则输出“否”，并停止。

3）否则$n$加2，返回第2步。

4）输出“是”，并停止。

这样一个程序具有算法的哪些特性——输入、输出、精确性、确定性、有穷性、正确性、通用性？其中哪些特性有赖于哥德巴赫猜想的正确性（数学家们尚未得出哥德巴赫猜想正确性的证明）？

[1]带标记S的习题表示书后附有答案。

## 1.2 表述算法的伪代码

尽管有时候完全可以用平常的语言来表述一个算法，不过计算机科学家们还是喜欢用**伪代码**（pseudocode）来表述，因为伪代码表述精确、结构清晰、通用性强。之所以称为伪代码，是因为它与实际的计算机语言代码，例如C++和Java，很相像。伪代码有多种版本。有一点它与实际的计算机语言不同，在实际的计算机语言中必须注意分号、字母的大小写、专用词等等，而伪代码的各种版本都可以用，只要它的命令不含糊和形式上相似，即使语法上不严格也不要紧。这一节讲述伪代码。

我们的算法是由包括标题、简要说明、输入和输出参数，以及包含该算法指令的函数等内容组成的。可以把找出三个数中最大值的例1.1.1写成算法如下：

**算法1.2.1 找出三个数中的最大值**。本算法找出$a$、$b$、$c$三个数中的最大值。

```
 Input Parameters:    a,b,c
Output Parameter:     x

max(a,b,c,x) {
        x = a
        if (b > x) // 若b大于x，更新x
            x = b
        if (c > x) // 若c大于x，更新x
            x = c
}
```

函数的第一行包括该函数的名称，随后是函数的参数（在圆括号里）。参数表示函数可用的数据、变量、数组等等。在算法1.2.1中参数是三个输入值$a$、$b$、$c$，以及输出参数$x$，$x$被赋值为三个输入值中的最大值。被函数执行的语句由大括号括起。

在**if语句**

```
if (condition)
    action
```

中，如果*condition*（条件）为真，*action*（动作）就执行，并且将控制传递给*action*后的语句。如果*condition*为假，*action*就不执行，控制立即传递给*action*后的语句。如上所示，我们用缩排来表示组成*action*的语句。并且，如果*action*由多条语句组成，则把它们括在大括号里。在if语句中，多语句的*action*例子是：

```
if (x ≥ 0) {
    x = x + 1
    a = b + c
}
```

条件语句的另一种形式是**if else语句**。在if else语句

```
if (condition)
    action1
else
    action2
```

中，如果*condition*为真，就执行*action1*（而不是*action2*），并且把控制传递给跟随在*action2*后面的语句。如果*condition*为假，则执行*action2*（而不是*action1*），并且把控制传递给跟随在*action2*后面的语句。如果*action1*或*action2*由多条语句组成，那么就应把它们括在大括号里。

两个正斜杠符号//表示一个**注解**的开始，一直延续到行末。算法1.2.1中注解的例子是：

```
// 若b大于x，更新x
```

注解并不执行，只是为了帮助阅读者理解算法。

算法表示中，保留词（例如，if）用常规字体表示，而用户自选的词（例如，函数名*max*，还有变量*x*等）用斜体表示。

**return语句**

```
return x
```

结束一个函数，并把*x*的值返回给这个函数的调用者。例如，对算法1.2.1中三个数中的最大值可以不用输出参数来存放，而把算法写成：

```
max(a,b,c) {
    x = a
    if (b > x) // 若b大于x，更新x
        x = b
    if (c > x) // 若c大于x，更新x
        x = c
    return x
}
```

语句

```
return
```

只是用来结束函数。如果没有return语句，函数在最后一个大括号前就会结束。

我们使用普通的算术运算符+、-、*（表示乘）、/，关系运算符 == （等于）、!=（不等于）、<或≤（小于或小于等于）、>或≥（大于或大于等于），及逻辑运算符&&（与）、||（或）、!（非）。注意，= 是赋值运算符，而 == 是等于运算符。

**while循环**被写成

```
while (condition)
    action
```

如果*condition*为真，则执行*action*，并且重复这个序列：就是说，如果*condition*仍为真，就再一次执行*action*。这个序列一直重复，直到*condition*变假。这时控制直接传递给*action*之后的语句。如果*action*由多条语句组成，则把它们括在大括号里。算法1.2.2使用while 循环找出一个数组中的最大值。设*s*是一个数组，*s.last*是这个数组中的最后一个下标。

**算法1.2.2 使用while循环找出一个数组中的最大值**。本算法找出数组*s*[1], *s*[2],…, *s*[*n*]中的最大值：

```
  Input Parameter:  s
Output Parameters:  None

array_max_ver1(s) {
```