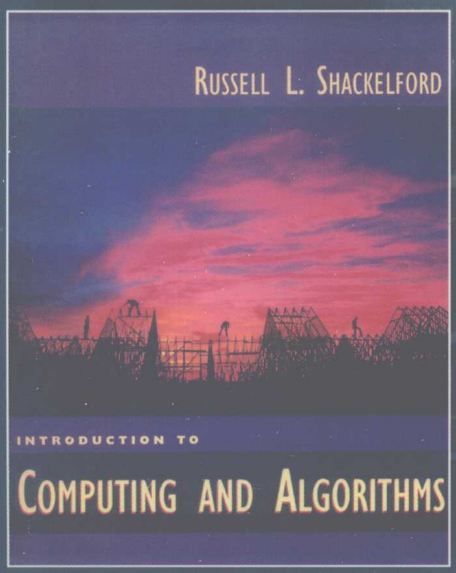
国外计算机科学教材系列

# 计算与算法导论

## Introduction to Computing and Algorithms




[美] Russell L. Shackelford 著
章小莉 孙厚琴 汪永好 等译

PEARSON Addison Wesley
PHEI
電子工業出版社
Publishing House of Electronics Industry
http://www.phei.com.cn

国外计算机科学教材系列



# 计算与算法导论

## Introduction to Computing and Algorithms

[美] Russell L. Shackelford 著

章小莉 孙厚琴 汪永好 等译

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

## 内容简介

程序设计是计算机专业学生学习的主要方向，然而，本书作者认为，算法的分析与构建比编程本身更重要，只有很好地解决了算法问题，才可能编写出好的程序。为此，本书分三个部分讨论了计算与算法的问题。第一部分主要回顾了西方历史上各种社会范式的发展，使读者可以了解科学的发展、社会的进步与人类对各种思维范式的研究紧密相关。第二部分概述了用于实现算法的伪代码中的结构和组件、原子基本数据和操作、过程、函数、参数和递归等各种知识，还介绍了查找、排序、优化等算法，此外，关于面向对象范式、正确寻址、正确估算算法的资源成本等也在本部分有专门的章节介绍。第三部分的目标是帮助读者了解什么样的问题能用计算机解决，区分并发与并行的概念，同时进一步讨论了如何将算法与实际问题相关联，并给出了近50年来的各种编程范例。

本书适合于各类院校的学生用做计算机知识入门课本，也是喜爱编程的人们培养分析问题能力的最佳参考资料。



**图书在版编目（CIP）数据**

计算与算法导论/（美）沙克尔福德（Shackelford, R. L.）著；章小莉等译. -北京：电子工业出版社，2003.11
（国外计算机科学教材系列）

书名原文：Introduction to Computing and Algorithms

ISBN 7-5053-9298-0

Ⅰ.计… Ⅱ.①沙… ②章… Ⅲ.电子计算机-计算方法-教材 Ⅳ. TP301.6

中国版本图书馆CIP数据核字（2003）第100113号

责任编辑：赵红燕

印 刷 者：北京兴华印刷厂

出版发行：电子工业出版社 http://www.phei.com.cn

北京市海淀区万寿路173信箱 邮编：100036

经 销：各地新华书店

开 本：787 × 1092 1/16 印张：19.25 字数：556千字

版 次：2003年11月第1版 2003年11月第1次印刷

定 价：29.00元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换；若书店售缺，请与本社发行部联系。

联系电话：（010）68279077。质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

# 出 版 说 明

21世纪初的5至10年是我国国民经济和社会发展的重要时期，也是信息产业快速发展的关键时期。在我国加入WTO后的今天，培养一支适应国际化竞争的一流IT人才队伍是我国高等教育的重要任务之一。信息科学和技术方面人才的优劣与多寡，是我国面对国际竞争时成败的关键因素。

当前，正值我国高等教育特别是信息科学领域的教育调整、变革的重大时期，为使我国教育体制与国际化接轨，有条件的高等院校正在为某些信息学科和技术课程使用国外优秀教材和优秀原版教材，以使我国在计算机教学上尽快赶上国际先进水平。

电子工业出版社秉承多年来引进国外优秀图书的经验，翻译出版了“国外计算机科学教材系列”丛书，这套教材覆盖学科范围广、领域宽、层次多，既有本科专业课程教材，也有研究生课程教材，以适应不同院系、不同专业、不同层次的师生对教材的需求，广大师生可自由选择和自由组合使用。这些教材涉及的学科方向包括网络与通信、操作系统、计算机组织与结构、算法与数据结构、数据库与信息处理、编程语言、图形图像与多媒体、软件工程等。同时，我们也适当引进了一些优秀英文原版教材，本着翻译版本和英文原版并重的原则，对重点图书既提供英文原版又提供相应的翻译版本。

在图书选题上，我们大都选择国外著名出版公司出版的高校教材，如Pearson Education培生教育出版集团、麦格劳－希尔教育出版集团、麻省理工学院出版社、剑桥大学出版社等。撰写教材的许多作者都是蜚声世界的教授、学者，如道格拉斯·科默（Douglas E. Comer）、威廉·斯托林斯（William Stallings）、哈维·戴特尔（Harvey M. Deitel）、尤利斯·布莱克（Uyless Black）等。

为确保教材的选题质量和翻译质量，我们约请了清华大学、北京大学、北京航空航天大学、复旦大学、上海交通大学、南京大学、浙江大学、哈尔滨工业大学、华中科技大学、西安交通大学、国防科学技术大学、解放军理工大学等著名高校的教授和骨干教师参与了本系列教材的选题、翻译和审校工作。他们中既有讲授同类教材的骨干教师、博士，也有积累了几十年教学经验的老教授和博士生导师。

在该系列教材的选题、翻译和编辑加工过程中，为提高教材质量，我们做了大量细致的工作，包括对所选教材进行全面论证；选择编辑时力求达到专业对口；对排版、印制质量进行严格把关。对于英文教材中出现的错误，我们通过与作者联络和网上下载勘误表等方式，逐一进行了修订。

此外，我们还将与国外著名出版公司合作，提供一些教材的教学支持资料，希望能为授课老师提供帮助。今后，我们将继续加强与各高校教师的密切联系，为广大师生引进更多的国外优秀教材和参考书，为我国计算机科学教学体系与国际教学体系的接轨做出努力。

电子工业出版社

# 教材出版委员会

# 译 者 序

20世纪最重要的发明是计算机，从表面上看，21世纪计算机和网络的不断应用将完全改变人们的生活方式，但是我们是否想过：今天的基因工程、生物医学研究、太空探测、每日天气预报等，都与计算机科学有着密不可分的关系，计算机不仅从形式上改变了人们的生活，而且从思维的深处改变着人类了解世界、研究世界的模式。

这些说起来那么神奇，与普通人离得非常遥远，可是翻开本书，从第1章、第2章短短的字里行间，我们就会了解我们每一个人都能充分理解人类进程的发展，并且真实感受和参与了这场社会变革，而且随着社会的变革，我们自觉与不自觉地改变了自己的思维——因为时代的思维范式确定了我们的行为。

那么，我们这个时代的思维范式是什么？就是计算范式。它复杂吗？难以理解吗？不，一点也不，只要把思维的方式与真正的实现分离开来看，它不过是一种与菜谱、游戏规则相类似的一组操作指令表，透过建立这些指令表的思维，不仅可以理解计算机的工作原理，而且可以学会建立自己的指令表，学会指挥计算机工作的思维。其实我们每天都在执行很多指令序列，只是从没有用专业术语即程序来称呼这些指令序列，我们从没有认真考虑过它们应该属于计算范式的思维模式。

本书回顾了西方历史上各种社会范式的发展，以及伴随范式而出现的科技进步、社会变革，并展望了现有思维范式的未来前景。其次用一种伪代码语言带领读者开始新思维的建立，并列举了许多常见算法的实现，以及计算机能力的局限性。

本书是为所有正在用计算机和打算用计算机的人们编写的，它虽然是一本理论类的书籍，但它与实际操作只有一步之遥。读者只要掌握任何一门计算机语言，就可以将本书中的例题稍加改造而真正实现。同时，本书也是一本计算机科学的入门书籍，任何致力于培养计算机科学工程师的院校都可以选用本书作为教材。

本书的翻译是集体劳动的成果。翻译人员有：杨东旻、章小莉、孙厚琴、汪永好、赵慧麟；此外，还得到了李正阳、王艳红、朱德芳、王玉琴、张宇、范永斌、徐日、周韩、宋燕红、曹驰、李珊、曹长宏等人的大力支持。

由于译者水平有限，难免有错译或不当之处，敬请读者批评指正。

# 前　言

## 概述

本书和它代表的入门课程是我们在Georgia计算机技术学院（Georgia Tech）进行计算机教育努力的结果，与此同时达到了两个目标：

1. 修正众所周知的计算机科学入门课程中存在的一些问题。
2. 满足各种需要计算机基础知识的学生的要求。

我们的努力已经被我们学校所在地区的人们广泛承认：使得仅在两年之内入学的新生人数从100人暴涨到1400人，现在该课程也被指定为大学教育中的核心课程。

因此，通过快速增长的入学人数，我们在所需的计算机入门教育新方法和班级管理技术方面获得了经验。顺着这条路，我们建立了跟踪学生的能力和学生对课程看法的过程，根据结果，我们更多地知道我们努力的效果比别的方式更好，这一追踪过程使我们能洞察新课程的开设将要冒多大的风险，也更相信获得的成果，事实上这是一个正面意义的成果。

## 本书读者和先行知识

本书的读者是各种学院和大学的广大学生，从我们过去5年的经验来看，所有教育和学习课程表上列有计算机科学、工程科学、自然科学、社会科学、数学、管理学、结构学为主课的学生都可成为本书的读者。我们仅缺乏对学习古典文学和健康科学的学生的教学经验，因为Georgia Tech不提供这些学科的教育课程。

学习本书知识不需要读者受过任何大学教育，但需要具有高中基础教育阶段的代数学基础和独立思维的能力。

## 本书使用方法说明

事实上，最近的10年清楚地显示计算机已经彻底地影响了科学、商业和通信。在这一点上，计算机与其他学科不同，计算机至少以两种方式影响着每个人。在工具类方面，计算机几乎正改变着人类所统治的每个领域。在概念方面，计算机正提供思维框架，这构成了实际上各个学科的尖端研究的基础。计算机无论是作为工具，还是它的思维方式都是当今新的社会范式的基础。这种新的范式正席卷整个现代社会。

我们坚信计算机学科的教育应该扩大它的视角，以便在社会提升计算这个新角色的地位。对于全世界来说，C++与Java之间和UNIX与Windows之间的不同，是一件小事。相反，对人类知识和理解力的历史来说，广泛地接受现象的算法模型是具有非常大的影响力的事情。在整个自然科学和行为科学中，艺术现在的形式是用算法术语表达的模型化现象，它生产了算法可以模拟测试和优化的这些模型。算法为理论和经验的学习提供了新的基本“游戏规则”基础。

我们坚信所有提供高质量教育的学院培养的学生应该接受算法思想的基本方法训练，也有经验表明，他们也应该将该课程添加到思维模式建立的训练中。本书中面向算法过程的建立不仅是计算机科学的主要内容，而且为广大的自学学生提供了“21世纪的基础知识”。计算机将导致人们生活现状的改变。未来受过良好教育的人们，需要知道计算机内部的思想和局限性。为此，我们的方法有三个主要目标：提供该领域的入门教育，提供该领域的概念内容和软件训练，以及为学生打好编程基础。

## 目标1：提供领域的入门教育

与其他学科课程不同，传统的计算机科学课程没有对学科的概念和智力基础的实质性概述，相反，总是一开始就让学生在没有任何基础的情况下“编写程序”。

与计算机相关的入门课程应该介绍些什么内容呢？一种可能是：入门课程应该给学生提供一个浏览计算机科学领域中各子学科的机会（例如在第四周必须了解操作系统）。我们不是沿用这样的老方法，因为我们的目的是使入门的学生理解计算机的主要思想，不是当前计算机科学各个组成部分的内容。我们发现，对计算机科学不为主课的学生来说，“浏览”并不能使他们建立起基本概念（而且可能更糊涂）。

我们采取的方法是认为计算机科学领域中核心的概念是算法，因此，我们把算法作为学习的主体。这种做法呼唤对基本思路的介绍，包括给一年级新生量身定做的基本算法结构和一定的理论基础。

## 目标2：提供概念的介绍和软件的训练

计算机的巨大冲击使得全体大学生都必须掌握计算机的基本使用技能。在许多大学，已经出现对计算机使用和标准软件应用相关的基本入门学习的需要。通常，这一需求导致了将计算机作为工具的非编程课程的出现（例如，教育学生使用字处理软件和电子表格软件等）。

我们坚决同意入门类计算机课程应该包含这些内容，但我们不同意这些内容就是全部。以我们的观点，“应用程序的使用”着重的是技术问题，因此它可以放在实践类课程中完成，但它缺乏主要内容，因而对大学水平的课程来说，将它作为基础课程是否合适是值得深思的问题。对我们来说，大学中学习怎样使用电子表格获得的学分无疑等价于电子工程系的学生学习焊接工艺或机械工程技生学习焊接工艺类课程获得的学分。

因此，我们开设的“计算机入门”课程具有交叉类学科的特点，当讲课与作业的主要内容是使用各种标准应用软件时，学校教与学的重要内容是从概念上构建和分析算法。本书提供教与学的训练，我们坚信它与不追究实现细节的各类大学课程相似。另一方面，实验自然有它的不同，要以任务研究和当地的计算机资源为基础。我们提供的实验训练要求以通用PC和基于PC的软件为操作环境，且具备经由网络实现对UNIX资源的访问。在原理上，它能在局域网环境中得以实现，书中实验训练内容由五大模块组件构成：

- 通信工具和设备。介绍电子邮件和新闻组、文本和图形编辑、桌面出版系统和HTML主页的创建。时间为4周。
- 数据处理工具和设备。介绍数据库、电子表格和解方程式软件。时间为4周。
- 问题求解。主要介绍解决特定学科问题的工具和设备的使用。时间为3周。
- 体验编程。第二阶段可称为“编程入门”的序言，它介绍Java的基本知识，包括构建组件和快速产生原型，用模仿和娱乐的方法进行教学。时间为3周。
- 实验训练估算。“告诉我你可以干什么”，基于实验的最后测验，安排在课程的最后阶段。

有关实验训练的更多信息和支持材料可以从面向用户的出版网站（参见补充资料）获得。

### 目标3：为编程打基础

教授传统编程入门课程的老师们的抱怨出奇地相似：在这类课程中，学生不理睬设计和实现的训练，怀疑文档的价值，不追求程序设计对问题的抽象，对答案“不求质量”。所有问题在于学生完全不理睬教师关于“编写”好程序应该做什么的教诲，这些及类似的教与学的问题无所不在。以我们的观点看，所有这些问题是关于编程的传统入门方法自然导致的结果，（a）因为它们使设计和实现问题模糊不清；（b）过早地让学生面对程序的实现任务；（c）课堂上一般过于强调特定语言环境中的结构和技术的实现。

我们坚信让学生在狭隘的特定语言环境（因此难以掌握有效的抽象）中积累经验是传统方法的目标，因为这种方法同时将学生放在编程环境中，要求他们必须解决好算法执行需求问题（因此，暗含着强调学生对实现而不是设计更关心），其结果预示着：当学生“在与编译器奋战至夜里2点的时候，他们很快就失去了对设计与抽象的全部感觉，所有的一切都被“在这里放个分号，在那里放个逗号，看工作吗”的调试过程所取代。我们坚信要克服这种现象，需要把实现与调试和设计与抽象分开来学习。经验告诉我们：如果没有将这两方面的问题区别开来，学生将过分专注于后者，其结果是绝对难以让他们关注前者，分散了他们学习前者的精力。

因此，我们尽力做到在简单的环境（写伪代码和画图方式）中学习抽象和设计的基本问题，在简单环境中，由于学习中不接近编译器，因此学员自身的能力决定了最后自己的水平。关于这一点，清晰地表明设计和实现比语言编译和算法的精确度更重要。一旦学生打下了设计和实现的基础，我们才加上必须满足计算机需求的学习。我们坚决主张在第一阶段课程（计算机入门）中，学生应该将精力集中在掌握逻辑抽象和设计的学习，以便他们进入第二阶段学习（编程入门）时了解他们正在努力干什么。第一阶段是基本方法概念和技术介绍，第二阶段的高级课程主要是学习有效设计、实现、测试和调试技巧与策略。简而言之，我们认为给学生打下编程基础的秘密是直接让他们获得这些准备。我们相信，与之相反的做法（特别是当在入门课程学习C++和Java这样的复杂语言时）都将是不明智的，只会取得不好的效果。

## 课程内容

本书共分三大部分，它们是：

### 第一部分：计算视角

这一部分的内容不含技术。第1章介绍了西方历史和科学中的计算问题，其目的是让学生了解计算的视角，以及人类关于知识和我们生存的世界之间的关系。在这一章中要触发学生对大范围而不仅是日常技术相关问题的好奇心。这一目标支持刺激学生进行反思，关注“大问题”的传统教学目标。紧接着的第2章高度概括了什么是算法、算法的组成和与算法术语相关的问题。

#### 内容选择

后续各章的学习可以不需要第1章做基础，因此，第1章可以不讲或作为选修材料。但是，从我们的经验来看，大三或大四的学生通常反映这一章的内容（对应的第一堂课）最能刺激他们整个大学期间的学习。第2章内容是后续课程的基础，因此为必修课。

## 第二部分：算法实现工具

这些章占据本书的大部分篇幅。在第二部分，快速全面概括了算法结构和相关的伪代码组成。第3章介绍基本数据和操作。第4章介绍过程抽象，包括过程、函数、参数和递归。第5章介绍整个的数据结构，包括记录、数组、链表、树和图。第6章介绍常用算法，如查找、遍历、排序和优化算法。第7章介绍与结构相关的面向对象范式。第8章解决改错和验证问题。第9章让学生理解和估算算法成本和复杂度的方法，并掌握对它们的估算，正确区别合理的和不合理的性能。掌握这些章节的知识之后，实际上学生已经准备好学习任何一种编程语言，已经完全掌握了软件工程和计算机科学理论的基础知识。

**内容选择**

第3~6章为必修课，一章一章知识相互衔接。第7章介绍面向对象范式，对那些还不打算学习这种范式的人来说，可以跳过本章不学。对于学习者来说，第7章可以直接在第6章之后学，也可以在第9章之后学。我们试验过在课程的不同阶段讲授它，目前是在第6章之后讲授它。经验表明过早学习这一章（如在第5章之后）效果不佳。但是如果放在第9章与第10章之间，则是可行的。将这部分内容移至第9章之后的主要优点是这种安排为排序算法的练习提供更多的解决方案，因此更利于学生消化所学内容。

## 第三部分：计算的局限性

这一部分的目的是帮助学生理解计算机可以做什么和不可以做什么。第10章讲述并发和并行，告诉学生如何通过并行突破可能的性能限制的逻辑关系。第11章扩展了第9章中的算法到问题概述，向学生介绍了确定的、不确定的和NP完全问题的概念，也更进一步从实践和理论的角度概括了计算机能解决的各种问题。最后，第12章重回到第1章历史发展的主题，同样给出了各种西方历史范式的发展和近50年来的各种编程范式的发展。

**内容选择**

对寻找方法让学生正确理解计算机能做什么和不能做什么的课程来说，这一部分是必修内容。对于只打算让学生学习算法和相关结构编程基础的课程来说，这一部分可以选修。很显然，我们的愿望是让学生从一开始就打下良好的基础，因此，这一内容作为必修课。同时，我们认识到有人可能不赞同这种课程目标和做这样的优先处理，另一些人可以希望用一整学期来学习前两部分。

## 给老师的话

本书在下面几个方面与一般的计算机科学入门课程用书不同：

- 对入门课程来说，它认为应该把现代编程语言的教学转变成“更多”理论的教学。

当20世纪60年代中期第一门关于编程的入门课程出现时，这一课程是合理的必修课。那时，今天我们了解的计算机科学尚不存在，计算机就意味着编程，编程语言本身很简单，这一课程主要是教授仅有的几个问题（如赋值、数据操作、原子数据类型、数组、if-then-else语句、带参数的过程、文件操作和简单的文本I/O格式），简言之，我们能够教全部吗？大多数时候，我们肯定能。

在最近的30年中，关于计算机，几乎所有的内容都发生了变化，现在当我们考察编程入门训练时，我们不仅能找到原始课程中的八大术语，而且还存在有更多的内容。现在，课程内容有指针、链表、树、递归、结构设计、交互式调试器、交互式程序、人－机接口、屏幕图形、大程序、软件

工程、复杂算法、多种语言、应用软件的应用，还有面向对象的设计和便携式程序，并且按照三个因素在增长。更进一步看，我们发现上述所提的内容每一项都比以前更复杂，实际上，教和学受到的挑战被大量因素影响，而且难度越来越大。

我们不是轻易决定要让学生上这门课，因为可以看到高年级的学生已经养成了很不好的习惯，他们对设计没有良好的认识，只追求仅有的软件开发的感觉。错，既不在学生，也不出自教师，而是因为基于语句的设计自身的问题。就基本入门教材而言，它含的内容太多了，很难在很短的时间内教完。其结果是教得越少，学得就越有效。学生学得很仔细，他们能用好指针，能遍历数据结构，但是他们从来没有以某种方式“获得”我们要他们“得到”的知识——抽象、设计、实现和估算的综合能力。我们注意到我们从来没有将这些内容放在一起教，学生也从来没有将它们放在一起学和用。大部分时间我们没有这么讲，是因为课程中没有足够的时间来做这件事。

- 为了获得时间，在真正的编程语言和真正的编程学习过程中，我们选择跳过许多烦人的内容细节不讲。

现代编程语言有几个烦人的方面。其中最烦人的是它们的语言和语义非常精密，使得实际用它们进行编程完成某个练习时往往受挫，特别是对初学者来说。实际上，为了将学生从成功编译和执行程序所需的要求中分离出来，专心我们要他们掌握的原理的学习，我们用伪代码语言——不需要任何实用处理器的编译或执行支持——来完成这种分离。事实上，学生不再花难以计数的时间来与编译器较劲，从而节省了大量的时间和精力。

另外，由于仅将自己的精力集中在基本结构（或多或少），不理睬各种与实际语言的交互，也使我们节省了大量的时间。例如，不管复杂的I/O命令，只用简单的输入和输出语句，至少可以节省一周半的时间。只用单个循环而不是三或四重循环可以节省一周时间。只用单一分支语句可以节省半周时间。实际上，当我们努力让学生专心于算法的理解和学习时，我们就把大量非基础语言细节的学习限制为最简单的语法学习，省去了编译和执行不通过的烦恼，这意味着我们可以集中精力学习重要原理和它们的应用。我们知道学生会因一些难题而分心，所以我们去掉了大部分难题而没有牺牲抽象的功能。

- 给学生“更大的想像空间”。

计算机教育几乎与每个人有关，这一事实意味着我们不再在遇到高级决策课题时才开始重视主题的讲解，这样做意味着非主修计算机科学课程的学生不用面对语言，减少了语法和编译的困惑，这也意味着我们可以在低年级学生学习中实现计算机科学中关键原理的讲解。因为在入门阶段我们主要强调基本软件工程原理（而不是关心它们的实现），学生不会养成后面必须消除的坏习惯，因为我们早早地介绍的是绝对的原理和复杂理论的应用，学生从没有机会“害怕”理论，使他们将算法的完成看成是自然的和一般的问题。

强调使用强大的软件应用程序进行实践与面向算法的学与做的训练的组合，意味着学生通过抽象的算法原理和广泛应用于这些复杂应用所需的模块设计学会了使用这些原理。因为学生在入门课程中彻底从概念上掌握了复杂的算法结构，当他们进入后续的“编程入门”课程时，知道了他们正努力做的事情是什么，以及为什么这样做。因此他们将以更加稳健的步伐在编程课上取得进步，比以前在编程方面可能取得更大、更多的进步。

- 在面向结构与面向对象编程之间架起了桥梁。

作为许多编程解决方案中的一种可选方法的面向对象编程(OOP)范式是为了替代费时费力的结构化程序设计范式而出现的。OOP范式有更多的语言特性，它暗示着在算法设计上存在根本的不同。同时，世界上大多数东西是，并将继续是嵌入在结构化范式中，因此，我们面对的尴尬是继续使用众所周知的、有一定效果的和有用的结构化范式，还是转而使用初现的OOP范式，不再顾及大多数学生和这些未来的员工在实际使用它们时它们存在的缺点。

当然，我们坚信两者都不可取。一方面我们不能不理睬向OOP的发展，或通过Java向硬件独立的OOP的发展。我们坚信Java语言，与WWW相互协作使用，不久的将来将戏剧性地改变很多东西。同时，我们对面向结构方法正在走下坡路这一论点表示怀疑，特别是因为它可以当做OOP语言的基础来学习，也因为结构化方法确实在许多学生和雇员中拥有自己的地位。

现在，我们选择让学生既学习面向对象范式，又学习OOP范式，因此，关于过程和数据抽象、算法方法的章节打下了面向结构范式的基础，随后通过仅需的结构知识的学习，实现OOP方法基础入门，从而使学生具备使用Pascal，Ada，C和FORTRAN的能力。他们也肯定有“操作能力”，能快速地在OOP的Java和C++世界中游刃有余。对那些认为我们应该完全向OOP方向转换的人们，我们说“是的，但不完全是”。我们期望本书的未来版本会更多地在OOP方向上进行问题讨论，但我们坚信，忽视结构化范式的时代还不会到来，这是软件世界中的转变时代，这一版图书也在这方面有意做了努力。

## 参考资料

本书的参考资料如下：

1. 出版公司的网站，www.awl.com/cseng/titles/0-201-31451-7/，在那里可以找到早期关于“方法解释”方面的应用程序的实验材料，找到与本书相关的参考资料。
2. 教师手册。含有我们从实践中总结出来的特殊教学方法，以及例题的答案和建议的估算策略。
3. 作者的研究。包括各种教与学效果的跟踪结果和优化策略，更有在整个课程学习过程中通过email与我们合作的合作伙伴的工作成果。

## 致谢

仅仅五年中，在Georgia Tech大学我们尝试了一种新方法来完成计算机教育的入门课程，并使它成为了大学的核心课程。从五年前的角度来看，任何一种改变看上去都不讨人喜欢(如果不是不可能)。我的朋友和同学Peter Freeman，Richard LeBlanc和Kurt Eiselt大力支持和鼓励我走过了改变现存状况的烦人过程。实际上，他们是使改变得以实现的人。另外，Richard LeBlanc也积极参与了课程和内容的修正。

这个项目的核心思想是在众多人的帮助下完成的，许多本科学生充当了教学助手，这些助手在课程中为了实现每学期几百名学生从思想到工作过程的平稳过渡起了关键作用。他们大多工作很长时间、获得很少报酬，目的是使所在的学校更加完美。这些助手多得令人难以记住他们的名字，但要特别提到的有(按字母顺序)John Brewster，Mark Canup，Charlie Carson，Becky Carvin，Dameon Kindall，Brian McNamera，Jon Preston，Roy Rodenstein，Enda Sullivan和Brian Toothman，他们都对课程或书本的多方面做出了贡献，解决了其中的不少问题，并初步试用了本书的内容。另外，Jason Bennett，Stu Bernstein，Mike Brogdon，Theresa Browne，Paul Griswold，Jim Hudson，Steven Knickerbocker，Danny Lentz，Geoff Menegay，Kristen Schaffer，Emily Stretch和Adrian Yang也做

出了重要的贡献，使得我认识到入门课程对毕业学生与教员的有价值的改革。就本人而言，我和学生们热情和高兴地共同工作着，我以极大的热情向他们推荐本书。

我的工作是以广泛流行的 David Harel 的工作为基础的。我们认为他的书“Algorithmics: The Spirit of Computing”清晰地介绍了计算机科学的基本思想，原书的目的是给“有基础的人们和计算机专家”看的，我们努力采用他的训练方式对现代大学生进行培训，不管我们是成功还是失败，正是Harel的工作激发了我们进一步努力，我们从他那里学到了他推荐给我们的一些我们不知道的方法，从而为我们打开了一个激动人心的新世界。

将一门课程和它相应的实验写成一本书是一件复杂和耗费经历的工作。当我在Georgia Tech验证本书的初稿时，它获得了较高的评价。在准备本书的过程中，我从下列列出的人员那里得到大量无价的帮助，他们对原稿进行了评价和批评指正：耶鲁大学的 Roger M. Smith，Potsdam SUNY 的 Charles Marshall，Southern Maine大学的Stephen Fenner，Austin Texas大学的R. Philip Bording，Stony Brook SUNY的Peter B. Henderson，Idaho大学的John Dickinson，Central Florida大学的Niel da Vitoria Lobo，Stonehill 大学的 Ralph Bravaco，North Carolina A&T 州立大学的 Ray Hawkins。由于他们的努力，本书的内容发生了巨大的变化。我从他们几个坦率的有兴趣的建议和与我合作的工作中受益匪浅。

Addison Wesley Longman 的许多人也在本书的写作过程中给我提供了有益的帮助，帮我完成了工作，其中 Dorothy Moore，Susan Hartman，Amy Willcutt，Julie Dunn 和 Tom ZiolKowski 最值得一提。另外，编辑 Stephanie Magean 和 Bobbie Lewis 也帮我克服了不少文字上的问题。最后在我与 AWL 联系之前，我幸运地从 Debbie Berridge 处取得有价值的支持和意见。

Russell Shackelford
Atlanta, Georgia

# 目　录

## 第一部分　计算视角

## 第二部分 算法工具