国外经典教材·计算机科学与技术

PEARSON Prentice Hall

Introduction to C and C++ for Technical Students: A Skill-Building Approach

Second Edition

C和C++基础教程与题解（第2版）

(美) Timothy S. Ramteke 著

施平安 译

Pearson Education

清华大学出版社

国外经典教材 计算机科学与技术

C 和 C++基础教程与题解

（第 2 版）

（美） Timothy S. Ramteke 著

施平安 译

清华大学出版社

北 京

内 容 简 介

本书既介绍了面向过程程序设计，又介绍了面向对象程序设计，书中语法的介绍与面向对象的原理、实践、分析和设计紧密地结合在一起。每个单元包括正文、练习、实验、问答题和程序设计部分，此种安排颇具匠心，帮助读者完全理解每个单元的内容。

本书既可以作为大专院校计算机及相关专业的程序设计基础课程的教材，也可以作为 C 和 C++ 语言的自学教材。

图书在版编目(CIP)数据
C 和 C++基础教程与题解（第 2 版）/（美）拉姆特克（Ramteke, T. S.）著；施平安译. —北京：清华大学出版社，2005.1
（国外经典教材 计算机科学与技术）
书名原文：Introduction to C and C++ for Technical Students: A Skill-Building Approach Second Edition
ISBN 7-302-09955-3
Ⅰ.C… Ⅱ.①拉… ②施… Ⅲ.C 语言－程序设计－教材 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2004）第 121342 号

出 版 者：清华大学出版社 **地 址**：北京清华大学学研大厦
http://www.tup.com.cn **邮 编**：100084
社总机：(010) 6277 0175 **客户服务**：(010) 6277 6969
文稿编辑：汤涌涛
封面设计：久久度文化
印 刷 者：北京市世界知识印刷厂
装 订 者：三河市金元装订厂
发 行 者：新华书店总店北京发行所
开 本：185×260 **印 张**：39.25 **字 数**：1183 千字
版 次：2005 年 1 月第 1 版 2005 年 1 月第 1 次印刷
书 号：ISBN 7-302-09955-3/TP • 6843
印 数：1~4500
定 价：59.00 元

出版说明

近年来，我国的高等教育特别是计算机学科教育，进行了一系列大的调整和改革，急需一批门类齐全、具有国际先进水平的计算机经典教材，以适应当前我国计算机科学的教学需要。通过使用国外先进的经典教材，可以了解并吸收国际先进的教学思想和教学方法，使我国的计算机科学教育能够跟上国际计算机教育发展的步伐，从而培育出更多具有国际水准的计算机专业人才，增强我国计算机产业的核心竞争力。为此，我们从国外知名的出版集团Pearson引进这套“国外经典教材·计算机科学与技术”教材。

作为全球最大的图书出版机构，Pearson在高等教育领域有着不凡的表现，其下属的Prentice Hall和Addison Wesley出版社是全球计算机高等教育的龙头出版机构。清华大学出版社与Pearson出版集团长期保持着紧密友好的合作关系，这次引进的“国外经典教材·计算机科学与技术”教材大部分出自Prentice Hall和Addison Wesley两家出版社。为了组织该套教材的出版，我们在国内聘请了一批知名的专家和教授，成立了一个专门的教材编审委员会。

教材编审委员会的运作从教材的选题阶段即开始启动，各位委员根据国内外高等院校计算机科学及相关专业的现有课程体系，并结合各个专业的培养方向，从Pearson出版的计算机系列教材中精心挑选针对性强的题材，以保证该套教材的优秀性和领先性，避免出现“低质重复引进”或“高质消化不良”的现象。

为了保证出版质量，我们为该套教材配备了一批经验丰富的编辑、排版、校对人员，制定了更加严格的出版流程。本套教材的译者，全部来自于对应专业的高校教师或拥有相关经验的IT专家。每本教材的责编在翻译伊始，就定期不间断地与该书的译者进行交流与反馈。为了尽可能地保留与发扬教材原著的精华，在经过翻译、排版和传统的三审三校之后，我们还请编审委员或相关的专家教授对文稿进行审读，以最大程度地弥补和修正在前面一系列加工过程中对教材造成的误差和瑕疵。

由于时间紧迫和受全体制作人员自身能力所限，该套教材在出版过程中很可能还存在一些遗憾，欢迎广大师生来电来信批评指正。同时，也欢迎读者朋友积极向我们推荐各类优秀的国外计算机教材，共同为我国高等院校计算机教育事业贡献力量。

清华大学出版社
2004.03.20

国外经典教材·计算机科学与技术

编审委员会

译 者 序

C++是当前商业软件开发的主流语言，效率高而功能强大，用它不仅可以进行面向过程程序设计，而且可以进行面向对象程序设计。但是从语言教学的角度考虑，C是一种比C++更好的语言，因为C语言比较简单，学生可以轻松地掌握基本逻辑。另外，因为C语言比C++语言的推出时间早，并且C++只是扩展了C语言在面向对象程序设计方面的功能，因此，如果先学C然后再学C++，学生们也会觉得更自然一些。因此，本书同时介绍了C和C++。第1～10单元介绍了C语言的基本知识；第11～14单元介绍了C语言的高级主题；第15～21单元介绍了C++语言的中级主题；第22～26单元介绍了C++语言的高级主题。

很多关于C和C++的教材几乎完全介绍语言的语法，而不注重编写程序所需的基本逻辑技能。如果C程序员仅仅学习C++新增的语法，并不一定会成为优秀的面向对象程序员，但还是很容易用C++写出面向过程的代码。为了充分利用C++的好处，程序员必须考虑面向对象设计。原书作者在这一方面另辟蹊径，尽量浓缩语法学习时间，并将语法的介绍与面向对象的原理、实践、分析和设计紧密地结合在一起，从而获得更多的时间来学习编写程序所需的逻辑技能和基本概念。本书可以作为大专院校计算机专业和非计算机专业的程序设计基础课程的教材，也可以作为C和C++语言的自学教材。

本书的一大特色是，每个单元都包括正文、练习、实验、问答题和程序设计部分。正文部分解释新概念，并用例子加以说明；练习和解为学生提供额外的练习；实验部分运用计算机运行和测试程序代码；问答题部分巩固正文中讨论的重要主题；程序设计部分提供解决问题的实践。

全书的翻译是集体工作的结晶。施惠琼、施琳琼、柳赐佳、周莎莎、黄山松、戴寿杰、余坦克、汪阅东、李树杰、孙琳、洪道金和陈学等负责全书的翻译工作，蔡荣荣和陈建伟等负责全书的审校工作，施金庭、柳聿荫和王子兰负责全书的录入和排版工作。全书最后由施平安负责统稿。

在翻译过程中，我们对本书中出现的所有术语和难词难句都进行了仔细的推敲和研究，然而有些方面在译者本人的研究领域中也不曾遇到过，疏漏和争议之处在所难免，望广大读者提出宝贵的意见。

前　言

本书目标

从C过渡到C++

在C语言创建时期，过程构成了程序的基本构件块。这种程序设计风格称为**面向过程**的程序设计。在C语言中，过程称为**函数**。函数是一系列主要按顺序执行的语句集合，而且函数通常要调用其他函数。所有的C和C++程序都必须包含main()函数，因为main()是这类程序执行的起点。我们将用前2个单元来学习函数以及如何使用printf()和strcpy()等预定义函数。然后在第9单元，我们将自己编写除了main()外的其他函数。在这种早期使用函数编写程序的风格中，数据项与函数分离。函数可以在不用对数据做任何限制的情况下操作数据项。数据项也称为**变量**，第2单元介绍了数据项，研究了它们的属性。

在C语言推出之后，编写程序的方式从设计函数改变为设计对象。**对象**把数据项和函数组合成一个实体，从而限制和控制可以对对象的数据项进行的操作。对象能够模拟我们每天感知的真实世界，日常生活中的万事万物都可以看作对象。虽然人们不曾想过我们的世界是由许多具有某种行为的对象构成的，但那确实是我们的世界的构成方式。你是一个对象，我是一个对象，本书也是一个对象，我们碰到的万事万物都是对象。当你阅读本书时，我就发送消息给你。至于如何响应这些消息，那是你的事情。

这种全新的程序编写方法与面向过程的程序设计方法相对应，我们把它称为**面向对象程序设计**(object-oriented programming)，C语言使面向对象程序设计成为可能。创建C++的思想不是创立一种全新的语言，而只是在C语言中加入全新的功能，将它转变成一种面向对象语言，于是把这种新语言称为C++。这就使广大的C程序员在学习面向对象语言方面有一个良好的开端。因此，C++只是扩展了C语言在面向对象程序设计方面的功能。第3单元讨论了如何使用预定义的对象，包括发送消息给cin、cout和字符串对象。然后在第15单元，我们将编写我们自己的对象类型，即所谓的**类**。

遗憾的是，如果C程序员仅仅学习C++新增的语法，并不一定会成为优秀的面向对象程序员。我们仍然很容易用C++写出面向过程的代码。为了充分利用C++的好处，程序必须考虑面向对象设计。因此，第16单元和第17单元运用一种称为CRC(class-responsibility-collaborator，类-责任-协调者)卡片的简单而功能强大的工具，帮助大家理解如何完成面向对象设计。后续单元继续阐述C++的特点。每学习一个新特点，我们都将分析它如何使我们能够更好地模拟真实世界。这种分析也有助于我们用一种全新的观点来了解我们生活的世界。面向对象程序设计提供了一种与我们观察日常生活完全相同的程序编码方法。

为什么仍然需要C程序设计技能

大量代码仍然是运用普通C语言以面向过程的风格编写的。许多从C转换成C++的项

目,由于程序员没有受过面向对象分析与设计方面的正规培训而失败了。许多过程代码是用C++ 语法编写的,因此,仍然有大量用普通 C 编写的遗留代码,公司也需要程序员来维护大量的此类 C 代码。

为什么 C 变成一种比C++ 更好的选择还有其他原因。C++ 要占用大量资源,用C++ 编写的代码,其执行速度通常不如普通 C 代码快。对于许多对快速响应的要求很高的实时应用,C 通常是一种更好的选择。现在,处理器速度正在飞速提高,速度也许不会成为制约问题。但是在嵌入式系统中,诸如汽车或者洗衣机中的计算机,没有足够的资源来运行C++ 代码——没有操作系统,没有足够的 RAM,等等。在这些情况下,C 是一种比C++ 更好的选择。运用 C 的面向过程风格的程序设计也是 Unix 的基础。

我个人认为,在教授新学生时,C 是一种比C++ 更好的语言,因为 C 语言比较简单,有助于学生轻松地学习基本逻辑。从推出时间上看,C 语言比C++ 语言早,因此,如果先学 C 然后再学C++ ,学生们自然会觉得更容易些。对于 Java 也是一样的道理。我认为学了C++ 以后再学习 Java 会更容易些,因为 Java 在C++ 之后推出。然而与许多学生想像的不同的是,C++ 新增了许多特点,而不仅仅是能够用 cin/cout 替代 scanf( )/printf( )完成输入/输出。

一种全新的 C 和C++ 学习方法

C 和C++ 这两种语言在整个发展过程中,增加了许多功能。对于不熟悉 C/C++ 的学生,这些语言当前具有的功能也许太多了。因此,本书只是有选择地介绍和运用了其中的一些功能。例如,我相信,只要学会了如何使用基本的整型数据类型,就很容易学会与整数有关的其他数据类型。因而,全书只用了基本的整型数据。我们减少了学习语法的时间,而用更多的时间来学习编写程序所需的逻辑技能和基本概念。如果希望学习这些语言的方方面面,不应只选本书——还应有一本优秀的参考指南。

在程序设计的早期,由于语言的语法简单易学,我们用大量时间来教授基本逻辑。而现今,由于语言取得了很大的进步,我们不得不花很多时间来学习语法,几乎没有时间来发展基本的逻辑技能。因此,本书尽量简化语法,通过把更多时间花在逻辑上,使大家有更多的机会来练习各种逻辑,重新回到发展基本逻辑技能上来。例如,第 5 单元没有介绍任何新语法,而只是探讨了已知语句的各种安排。在一本书中讨论语言的所有方面是非常具有诱惑力的,但这样就不得不减少发展基本逻辑技能的课程量。所以本书着重介绍技能。

本书使用说明

各单元的组成部分

每个单元包括如下几部分:正文(Lesson)、练习(Drill)、实验(Experiment)、问答题(Question)和程序设计(Program)。某些单元还有一个称为附加主题(Additional Topic)的部分。练习、问答题和程序设计部分可以作为家庭作业在纸上完成,而不必使用计算机。实验和程序设计部分应该在实验室中完成,通常需要使用一台计算机和一个编译程序。这些部分应按什么顺序完成呢? 如果敢于冒险并且喜欢自学,则先完成实验和/或练习部分,然后

阅读正文。如果希望先解释概念,则先学习正文,然后通过做练习和实验巩固所学知识。既可以在做好练习和实验之前学习正文,也可以在做好练习和实验之后学习正文。图 A表示了这两种学习方法。

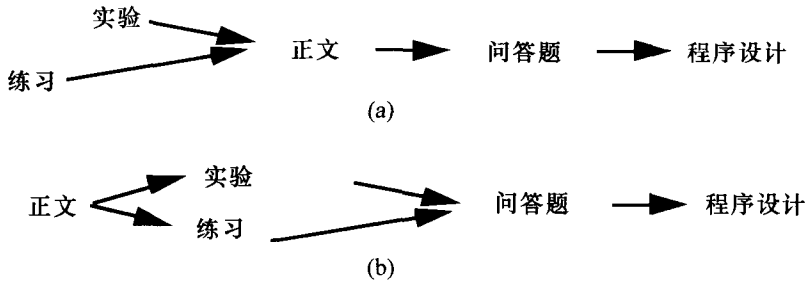

图 A　(a)试探性学习方法;(b)"示证性(show-me)"学习方法

正文。这是每个单元的常规内容。这里为大家提供有关阅读和学习的材料。这部分先解释新的概念,然后用例子说明它们。

练习及其解。这部分可以作为家庭作业用纸和笔完成,而不必使用计算机。在事先没有阅读材料之前,甚至学习正文之前,就应当能够完成这部分练习。复杂的思想被解剖成许多简单的思想,然后对这些简单的思想进行解释,因而要求反馈。首先在空白的纸上做出练习的答案。从练习部分的顶部开始,向下移动一张纸(或者自己的答题纸),让它盖掉第 1 道题的答案。阅读和研究介绍性材料,在答题纸上做出第 1 道练习的答案。然后,向下移动答题纸,揭开它的答案。将你的答案与给定的答案进行比较。改正所犯的任何错误,如果需要,重新检查你的答案,然后向下移动遮盖纸,让它遮盖掉下一道题的答案。重复这个过程,直到完成每道练习,并在进行下一道练习之前纠正所犯的任何错误。

在做下一道练习之前检查答案,可以预防不正确的思想,使你马上回到正确的思路上来。练习及其解部分可以如正文一样进行阅读。然而,最好是在空白的纸上动手做练习,查看答案对不对,然后改正它们。这样就可以在考试中犯相同的错误之前改正错误——没有人会注意。

实验。这部分必须在计算机上完成,并且可以在阅读正文和讲授有关单元的内容之前完成。只要阅读实验部分的导言段,就可以知道实验的目的,然后在计算机上输入并运行实验。将实验的输出写在一张纸上,这样就可以看到实验结果。如果能够运行实验,则很有可能正确地把程序复制到计算机上,应该不必担心输出结果是否正确。程序的输出几乎总是正确的,除非编译器有 bug,但这对于我们的程序几乎没有可能。

在运行实验之后,马上可以回答实验后面的问题。在回答了当前实验的所有问题之前,不要进入下一个实验。如果对某个实验的任何改变感到好奇,可以马上改变它,然后亲自观察会发生什么情况。另外,只有在明白了当前实验之后,才进入下一个实验。

问答题。应当能够在不用通过编译器运行它们的情况下,作为家庭作业完成这些问答题。你应当已经掌握本单元的有关知识,能够判断给定代码的效果。如果有时间,还可以通过编译器运行这些代码,但这只是为了检验答案是否正确,因为考试时一般不能使用计算机。本部分为你做好这方面的准备。

程序设计。既可以手工写出程序的解;也可以在计算机上运行它们,把它们的结果打印出来。本部分可以作为家庭作业,也可以在实验室完成,但是必须上交给教师。

各单元的学习顺序

图 B 给出了学习本书的三种主要方法。有些人可能只想学 C 语言,有些人可能只想学 C++ 的基本知识,还有些人可能只想学C++ 。我已经将 C 的所有 printf( ) 和 scanf( ) 语句改变成 cout 和 cin,cout 和 cin 是C++ 中的相应语句。那些只想学 C 语言的人,必须将所有的 cout 和 cin 语句转变成 printf( ) 和 scanf( ) 语句。第 1 单元讨论 printf( ) 语句,而第 3 单元讨论了 scanf( ) 语句。第 9 单元讨论函数时使用了 printf( ) 和 scanf( ) 语句,此外在第 12 ~ 14 单元的某些部分也使用了 printf( ) 和 scanf( ) 语句。在某些情况下使用 printf( ) 语句更容易,诸如打印存储了某个字符串的内存地址。C++ 中的某些概念可能不易理解。希望C++ 程序员也要学习 printf( ) 和 scanf( ) 语句,而 C 程序员也学习 cout 和 cin 语句。

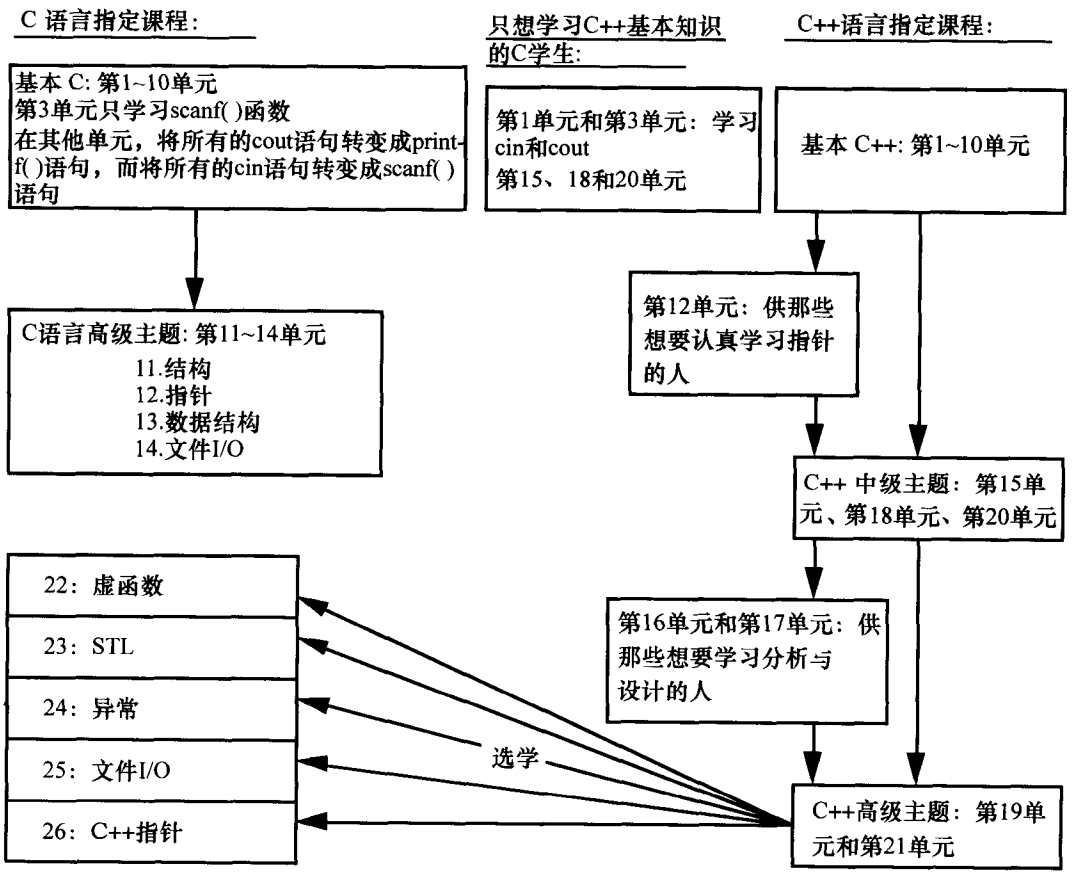

图 B 本书各单元的学习顺序

只要学过 C 语言,并且希望熟悉C++ ,建议学习第 15 单元、第 18 单元和第 20 单元。第 15 单元正式介绍面向对象程序设计、学习基本术语和编写简单的类。第 18 单元将学习构造

函数、析构函数和重载的基本知识。第20单元将全面介绍继承，即一个类派生于另一个类。

如果只想学习C++，则可以通过前10个单元打下坚实的基础。如果已经通过其他途径打下了该基础，也建议大家最好学习第3单元。第3单元简单地介绍了面向对象程序设计，而没有涉及第15单元阐述的术语。你可以不必学习第12单元，而直接学习本书的后续单元。只有在你需要全面学习指针的情况下，才需要学习第12单元。这是一个非常有趣的单元。你对某事物研究得越深，就会越运用得得心应手。因为C/C++程序员害怕运用指针，你也许正好需要学习第12单元。

面向对象分析与设计是编写良好的C++代码的关键。如果有时间，一定要学习本书描述的面向对象分析与设计方法。然后，准备学习第18单元和后续内容。第19单元讨论的是动态对象，可以根据需要跳过不学。而第21单元正式介绍了多态性概念，该单元是进一步学习本书其余内容的关键。然而，只要掌握了本单元，就可以任意选学第22～26单元的内容。只要学习了第19单元和第21单元，可以按任意顺序学习第22～26单元。

目　录

第1单元　函数和数据输出

正　　文

学习面向对象程序设计所需的步骤

本书详细讨论C和C++语言。ANSI(American National Standards Institute,美国国家标准协会)于1988年制定了C语言的标准,并于1999年再次修定C语言的标准。在1998年,ANSI还制定了C++语言的标准。本书中的所有代码(包括C代码)都可以用符合ANSI标准的C++编译器进行编译。

C++是C语言的面向对象扩展。因此,学习面向对象程序设计是本书的主要目的。我们从一开始就奠定面向对象程序设计基础(实际上是从第3单元开始)。在学习现代程序设计语言中使用的这种强有力的技术之前,必须了解对象是什么。简单地说,对象由两部分组成:函数(也称为方法)和数据项(也称为属性)。第1单元将学习函数,第2单元将学习数据项和变量,提供把这两个概念联系在一起所需的基础知识,同时为第3单元介绍对象做好准备。

我们之所以在第1单元研究函数,是因为函数构成了对象的半壁江山。我们重点讨论C的printf()函数,该函数通常用来在屏幕上显示输出。虽然全书一直都在使用它,但我们还会使用C++的cout语句来显示输出。因为printf()是一个函数,所以要了解它的基本行为。然而,本单元不会解释cout的工作原理,因为cout是一个对象,而我们直到第3单元才介绍对象。

虽然前3个单元将使用函数和对象,但这些函数和对象都是编译器提供的,我们只是运用它们而已。例如,我们将使用printf()函数和cout对象在屏幕上显示数据项,而不关心它们实际上是如何实现打印的。我们将使用它们来完成打印任务。然后在第9单元,我们将开始创建自己的函数,而在第15单元,我们开始创建自己的对象。现在,让我们开始讨论函数。

本单元主要学习printf()函数,说明该函数的用法。在本书的其余部分,该函数不会用得太多,而是用cout对象完成几乎所有的打印任务。那么,为什么要花时间学习printf()函数呢?原因之一是仍有许多C风格的代码必须维护,并且有许多应用采用C比C++更合适。另一个原因是printf()函数及其概念将会反复得到运用。学习printf()函数,还将使我们认识到基本数据类型一般指什么。如果主要目的是学习C语言,那么学完本单元之后,应当能够轻松地把cout语句转换成printf()语句。

什么是函数

在C和C++中,如果不用函数,甚至连一个简单的程序都写不出来。函数在我们的日常