普通高等教育“十一五”国家级规划教材 计算机系列教材

C语言程序设计教程

周彩英 主编

清华大学出版社

普通高等教育“十一五”国家级规划教材　计算机系列教材

主　编　周彩英
副主编　徐晶　贺兴亚　姜艺　潘钧

C语言程序设计教程

清华大学出版社
北京

内 容 简 介

本书是为普通高等学校学习C语言程序设计所编写的教材。全书共分为10章，按C语言程序设计的教学大纲并结合C语言程序设计等级考试大纲的要求，系统介绍了C语言及其程序设计的方法与技术。本书取材适中、结构合理、重点突出、难点分散、弱化不常用功能，并采用循序渐进的方法，极大地减轻了读者学习C语言的困难。

本书既可作为高等院校、高职高专非计算机专业“C语言程序设计”课程的教材，也可作为计算机专业程序设计课程的教材或参考书，对于参加C语言等级考试的同学具有一定的参考价值。

图书在版编目（CIP）数据

C语言程序设计教程 / 周彩英主编. —北京：清华大学出版社，2011.6
ISBN 978-7-302-25856-8

Ⅰ. ①C…　Ⅱ. ①周…　Ⅲ. ①C语言-程序设计-教材　Ⅳ. ①TP312

中国版本图书馆CIP数据核字（2011）第114757号

责任编辑：魏江江
责任校对：李建庄
责任印制：王秀菊
出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦A座
http://www.tup.com.cn　　**邮　　编**：100084
社　总　机：010-62770175　　**邮　购**：010-62786544
投稿与读者服务：010-62795954，jsjjc@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：北京国马印刷厂
经　　销：全国新华书店
开　　本：185×260　**印　张**：19.25　**字　数**：480千字
版　　次：2011年6月第1版　　**印　　次**：2011年6月第1次印刷
印　　数：1～4000
定　　价：29.50元

产品编号：041794-01

前　言

随便进入一家书店，来到计算机专柜，都可以看到琳琅满目的C语言书籍。在这种状况下写书，着实需要勇气和能力，因此书本的特色和实用性就显得尤为重要。考虑到现在高校中的学情和学时设置，本书的目标是力争成为最易懂、最专业、最时尚、最实用的C语言教材和参考手册。

本书是一本教材，适合于程序设计的初学者和想更深入了解C语言的读者。每个知识点的讲解都遵循由浅入深、循序渐进的原则，并以把问题讲清楚、讲明白、讲透彻，又不累赘为目标。同时抛弃了一些陈旧的和设计应用程序过程中不必过于纠缠的内容，试图把程序设计领域最新、最有价值的思想和方法渗透到古老的C语言中，赋予C语言一个焕然一新的面貌。本书在如下方面做了努力：

- 高标准。本书博采众家之长，自成一门课程的完整体系，对每个知识点的讲解、例题的选择与解题思路的分析都力求精益求精。也许它没有华丽的外表，但它有深刻的内涵；它不仅传播了知识信息，更着意进行科学思维与方法的点拨，能促使读者学会思考、学会分析、学会应用。
- 新角度。本书对C语言的教学内容进行了分层式处理，以求适用于不同学校、不同专业、不同学时数的读者。每章后的习题分为自我测评、基础练习和拓展训练3个部分，呈现出新的模式与跨越，蕴涵着对读者智能的深层开发。
- 大视野。教材立足课内，向课外拓展，知识面宽，信息量大，涵盖率高，既适合于课程的自学与辅导学习，也适合于各类相关的计算机等级考试，特别是“全国计算机等级考试”和“江苏省非计算机专业学生计算机应用能力水平考试”。
- 广思路。引导读者从多角度思考和切入问题，并向纵深发展；力图培养与规范读者驾驭思维、信息的能力，并激发他们寻找自己新的知识增长点。

全书共分10章，内容包括：程序设计基础、C语言入门、基本控制结构、函数、数组、指针、函数进阶和结构化编程、结构和联合、指针进阶以及文件。

本书注重教材的可读性和可用性。每章开头有学习目标提示，指导读者阅读；每章结尾都安排有实践活动和本章内容学习完成后的自我测评，以帮助读者整理思路，形成清晰的逻辑体系和主线；每章中的典型例题，由浅入深，强化知识点、算法、编程方法与技巧；习题以巩固基本知识点为目的，包括选择题、阅读程序写出运行结果、程序填空和编程题等各种计算机等级考试二级C语言考试中的常见题型；习题中还给出了拓展训练，将程序测试、程序调试与排错、软件的健壮性和代码风格、结构化与模块化程序设计方法等软件工程知识融入其中，力求做到从一点出发，向四周辐射，“心骛八极，思接千载”，从而编

织起知识结构的信息网络，达到思维培养的预期目标。

与本书同时配套出版的《C 语言程序习题解答与学习指导》提供了《C 语言程序设计教程》中全部习题的解答与分析、实验指导、课程考试与等级考试考点分析与例题精解，还配备有课程考试模拟试卷及计算机等级考试模拟试卷。相信本书一定会给读者以全新的体验和感受，希望读者读后能产生“众里寻他千百度，蓦然回首，那人却在灯火阑珊处”的共鸣。

学时建议：30 学时，可讲解第 1 章～第 5 章；48 学时，可讲解第 1 章～第 8 章；60 学时，可讲解第 1 章～第 10 章。因为全书采用分层式处理方式，所以每种学时建议中都未忽略 C 语言的精华部分。

全书的统稿工作由周彩英负责，第 1、7 章由周彩英编写，第 2、3 章由孙多编写，第 4 章由徐晶编写，第 5 章由贺兴亚编写，第 6 章由姜艺编写，第 8 章由潘钧编写，第 9 章由周旭东编写，第 10 章及附录由孟纯煜编写。在本次编著过程中，徐晶、贺兴亚、潘钧、姜艺和宋春来参加了全程的统稿工作，在此对他们的辛勤劳动表示衷心感谢，并对精心整理每章实践活动的陈洁表示衷心感谢，希望她在天国能感受到我浓浓的思念之情。

因编者水平有限，书中错误在所难免，恳请读者批评指正。

编者

于扬州大学计算机中心

2011 年 5 月

目　　录

第 1 章　程序设计基础

学习目标

1. 了解程序、程序设计、程序设计语言及其分类、语言处理程序等概念；
2. 理解算法、算法设计、算法的表示等概念，并初步学会根据问题设计算法；
3. 理解用于结构化程序设计的三种基本结构；
4. 熟练掌握用 N-S 图描述一个简单的算法；
5. 理解 C 语言源程序的结构；
6. 理解 main 函数及其他函数在 C 源程序中的位置及作用；
7. 了解 C 语言源程序的书写格式、程序设计风格；
8. 熟练掌握利用 TC2.0 或 Win-TC 系统编辑、调试 C 源程序的方法。

1.1　程序与程序设计语言

计算机程序（通常简称程序）是人们为解决某种问题用计算机可以识别的代码编排的一系列加工步骤，是一组指示计算机每一步动作的命令；计算机能严格地按照这些步骤去做，包括对数据的处理。程序的执行过程实际上是对程序所表达的数据进行处理的过程。

1.1.1　指令与程序

指令是规定计算机操作的一组字符，单独的一条指令本身只能完成计算机的一个最基本的功能，如实现一次加法运算或实现一次大小的判别。一台计算机所能执行的指令的集合称为指令系统或指令集。一台特定的计算机只能执行自己指令系统中的指令。虽然指令系统中指令的条数很有限，一条指令完成的功能也很简单，但一系列指令的集合却能实现很复杂的功能，这就是计算机功能强大奇妙之所在。一系列遵循一定规则和思想并能正确完成指定工作的计算机指令的有序组合就构成了程序。

1.1.2　程序设计与程序设计语言

1．程序设计

一个计算机源程序一般需要描述两部分内容：一是描述问题中的每个对象及它们之间的关系；二是描述对这些对象进行处理的规则。其中对象及它们之间的关系涉及数据结构的内容，而处理规则即求解某个问题的算法。因此，对程序的描述经常可以理解为：

程序=数据结构+算法

著明的瑞士计算机科学家沃思（N.Wirth）教授提出：“程序=数据结构+算法”，这里的数据结构是指数据的逻辑结构和存储结构，算法是对运算的描述。

一个设计合理的数据结构往往可以简化算法，而且一个好的程序应具有可靠性、易读性、可维护性等良好特性。

所谓程序设计，就是根据计算机要完成的任务，提出相应的需求，在此基础上设计数据结构和算法，然后再编写相应的程序代码并测试该代码运行的正确性，直到能够得到正确的运行结果为止。程序设计是要讲究方法的，良好的设计方法能够大大提高程序的清晰度和执行效率。通常程序设计有一套完整的方法，这一套完整的方法也称为程序设计方法学，因此有专家提出如下关系：

程序设计=数据结构+算法+程序设计方法学

程序设计方法学在程序设计中被提到比较高的位置，尤其对于大型软件的设计更是如此，它是软件工程的组成部分。

2．程序设计语言

程序设计语言（Programming Language）是一组用来书写计算机程序的语法规则。程序设计语言提供了一种表达数据与处理数据的功能，编程人员必须按照语言所要求的规范（即语法要求）进行编程。

在过去的几十年里，大量的程序设计语言被发明、被取代、被修改或组合在一起。尽管人们多次试图创造一种通用的程序设计语言，却没有一次尝试是成功的。之所以有那么多种不同的编程语言存在，其原因主要在于：编写程序的初衷各不相同；许多语言对新手来说太难学；不同程序之间的运行成本各不相同。因此，有许多用于特殊用途的语言只在特殊情况下使用。例如，PHP专门用来显示网页；Perl 更适合文本处理等。而 C 语言是被广泛用于操作系统和编译器开发（所谓的系统编程）的一种面向过程的通用高级语言，是面向对象的计算机语言（如 C#、C++、Java 等）的基础。

3．程序设计语言的分类

程序设计语言按照语言级别可以分为低级语言和高级语言。低级语言有机器语言和汇编语言。低级语言与特定的机器有关、效率高，但使用复杂、繁琐、编程费时、易出差错。高级语言的表示方法要比低级语言更接近于待解问题的表示方法，其特点是在一定程度上与具体机器无关，易学、易用、易维护。

1）机器语言

机器语言是计算机硬件能够唯一识别和执行的用二进制数表示指令代码的程序设计语言。机器语言执行速度很快，但通常人们编程时，不采用机器语言，因为它非常难于记忆和识别。不同机型的机器语言是不同的。

2）汇编语言

汇编语言的实质和机器语言是相同的，都是直接对硬件操作，只不过指令采用了英文缩写的标识符，是一种符号化的语言。与机器语言相比，汇编语言更容易识别和记忆。同样需要编程者将每一步具体的操作用命令的形式写出来，它的每一条指令只能对应实际操作过程中的一个很细微的动作，例如移动、自增等。因此，用汇编语言书写的源程序一般比较冗长、复杂、容易出错，而且使用汇编语言编程需要有更多的计算机专业知识。但汇

编语言的优点也是显而易见的，用汇编语言完成的操作不是一般高级语言所能实现的，源程序经汇编生成的可执行文件不仅比较小，而且执行速度很快。

3）高级语言

高级语言是接近人们习惯使用的自然语言和数学语言的计算机程序设计语言，独立于计算机。即使不了解机器指令，也不了解机器的内部结构和工作原理，也能用高级语言编写程序。高级语言主要由语句构成，有一定的书写规则，可以用语句表达要计算机完成的操作。由于高级语言有统一的语法，独立于具体机器，故便于人们编码、阅读和理解。

高级语言是一种既能方便地描述客观对象，又能借助编译器为计算机所接受、理解和执行的人工语言。例如，用于科学计算的 FORTRAN 语言；早期非常普及的 BASIC 语言；第一个用严格文法描述的 ALGOL 60 语言；适合底层程序（如驱动程序）开发的 C 及 C++语言；方便开发桌面应用程序的 Visual Basic 和 Delphi 语言；结合了 C 及 C++的强大功能以及 Visual Basic 的易用性的 C#语言等。

程序设计语言是软件开发的一个重要方面，其发展趋势是模块化、简明化、形式化、并行化和可视化。

1.1.3　语言处理程序

计算机只能执行机器语言程序，用汇编语言或高级语言编写的程序（源程序），计算机是不能识别和执行的。因此，必须配备一种工具，任务是把用汇编语言或高级语言编写的源程序翻译成机器可执行的机器语言程序，这种工具就是“语言处理程序”。 语言处理程序包括汇编程序、解释程序和编译程序。

1．汇编程序

汇编程序是把用汇编语言写的源程序翻译成机器可执行的目标程序的翻译程序，其翻译过程叫汇编。

2．解释程序

解释程序接受用某种高级程序设计语言（比如 BASIC 语言）编写的源程序，然后对源程序中的每一条语句逐条进行解释并执行，最后得出结果。也就是说，解释程序对源程序是一边翻译，一边执行，执行方式类似于“同声翻译”。解释程序在对源程序进行翻译时并不产生目标程序，因此，应用程序不能脱离其解释器，但这种方式比较灵活，可以动态地调整、修改应用程序。

3．编译程序

编译程序是将用高级语言所编写的源程序翻译成用机器语言表示的目标程序的翻译程序，其翻译过程称为编译。编译程序与解释程序的区别在于，它将源程序翻译成目标代码文件（*.obj），计算机再执行由此生成的可执行文件。一般而言，建立在编译基础上的系统在执行速度上都优于建立在解释基础上的系统。但是，编译程序比较复杂，应用程序一旦需要修改，必须先修改源代码，再重新编译生成新的目标文件才能执行，这使得开发和维护费用较大；相反，解释程序比较简单，占用内存少，可移植性也好，缺点是执行速度慢。

1.2 算　　法

1.2.1 算法的概念

1. 算法

为解决一个实际问题而采取的确定且有穷的方法和步骤，称之为“算法”。解决同一个问题，可能有不同的方法和步骤，即有不同的算法。例如，求 1+2+3+4+5+…+100，有的人可能先进行 1+2，再加 3，再加 4，以此类推，一直加到 100；而有的人则采取这样的方法：100+(1+99)+(2+98)+…+(49+51)+50=100×50+50=5050。当然，方法有优劣之分，有的方法只需执行很少的步骤，而有些方法则需要较多的步骤。一般来说，总是希望采用方法简单、运算步骤少的算法。因此，为了有效地进行解题，不仅需要保证算法正确，还要考虑算法的质量，选择合适的算法。同样的任务如果采用不同的算法来实现，可能需要不同的时间、空间开销，其效率往往也是不同的。一个算法的优劣可以用空间复杂度与时间复杂度来衡量。通常时间复杂度用于度量算法执行的时间长短；而空间复杂度用于度量算法执行时所需存储空间的大小。

2. 算法的特性

1）有穷性

一个算法应包含有限的操作步骤而不能是无限的，同时一个算法应当在执行一定数量的步骤后结束，不能陷入死循环。事实上“有穷性”往往指“在合理范围之内”的有限步骤。如果让计算机执行一个历时几年才结束的算法，算法尽管有穷，但超过了合理的限度，人们也不认为此算法是有用的。

2）确定性

确定性是指算法中的每一个步骤应当是确定的，不能含糊、模棱两可，也就是说算法不能产生歧义。特别是当算法用自然语言描述时更应注意这点。例如：“将成绩优秀的同学名单打印输出”就是有歧义的，不适合描述算法步骤，因为“成绩优秀”要求不明确，究竟是要求“每门课程都在 90 分以上”、还是“平均成绩在 90 分以上”或者是其他的什么条件等未作明确的说明。

3）有零个或多个输入

所谓输入是指算法执行时从外界获取必要的信息。外界是相对算法本身的，输入可以是来自键盘或数据文件中的数据，也可以是程序其他部分传递给算法的数据。可以没有输入，也可以有输入。例如：可以编写一个不需要输入任何数据，就可以计算出 5！的算法；但如果要计算任意两个整数的最大公约数，则通常需要输入两个整数。

4）有一个或多个输出

算法必须得到结果。没有结果的算法没有意义，结果可以显示在屏幕上或在打印机上打印，也可以传递给数据文件或程序的其他部分。

5）有效性

算法的有效性是指算法中每一个步骤应当能有效地执行，并得到确定的结果。

1.2.2　算法设计举例

程序设计人员的基本技能之一是必须会设计算法，并根据算法写出程序。因此，对于初学者来说，掌握一些常用算法是非常重要的。许多初学者常常把要解决的问题首先和程序设计语言中的语句联系在一起，影响了程序设计质量。设计算法和编写程序要分开考虑，在学习程序设计语言之前，就应该学会针对一些简单问题来设计算法。

【例 1-1】 有两个瓶子 A 和 B，A 中盛放酒，B 中盛放醋，现要求将其中的酒和醋互换，即 A 中盛放醋，B 中盛放酒。请设计一个交换两个瓶子中的酒和醋的算法。

分析：这是一个非数值运算问题。一般地，两个瓶子中的液体不能直接交换。要解决这一问题，最好的办法是引入第 3 个空的瓶子 C。这样，交换步骤可描述为如下算法：

（1）将 A 瓶中的酒装入 C 瓶中；

（2）将 B 瓶中的醋装入 A 瓶中；

（3）将 C 瓶中的酒装入 B 瓶中。

评注：可将 3 个瓶子看成是 3 个变量，瓶子中的酒和醋看做是变量中存放的值，这个算法就可以引申到交换两个变量的值。

【例 1-2】 请设计算法，实现如下问题的求解：输入 3 个数，输出其中的最大数。

分析：这一问题处理的对象是 3 个数，那么，首先需要有空间存放这 3 个数，定义 3 个变量 a、b、c，将 3 个数依次输入到 a、b、c 中。另外，因要求输出最大数，所以可再定义一个变量存放最大数，设该变量为 max。算法描述如下：

（1）输入 a、b、c；

（2）比较 a 与 b 的大小，将 a 与 b 中的大者放入变量 max 中；

（3）比较 c 与 max 的大小，将 c 与 max 中的大者放入变量 max 中；

（4）输出 max。

【例 1-3】 求 $\text{sum}=\sum_{i=1}^{6} i$，即 sum=1+2+3+4+5+6 的值。

方法一：可以用最直接的方法进行计算。算法描述如下：

（1）定义变量 sum；

（2）求 sum=1+2+3+4+5+6；

（3）输出 sum。

评注：这样的算法虽然正确，但不具通用性，太繁琐了。如果需要求 1~100 的和，则要书写的表达式太长，显然是不可取的。

方法二：利用等差数列求和公式计算。算法描述如下：

（1）定义变量 sum；

（2）求 $\text{sum}=\dfrac{6\times(1+6)}{2}$；

（3）输出 sum。

评注：这样的算法虽然效率较高，但前提条件是必须知道等差数列的求和公式。

方法三：把求和公式转化为 2 个简单公式的重复计算。算法描述如下：

（1）定义变量 sum，用于求累加和，定义变量 i 用于表示[1, 6]内的任一整数；

（2）令 sum 的初值为 0，i 的初值为 1；

（3）令 sum=sum+i；

（4）令 i=i+1；

（5）如果 i<7 则转（3），否则转（6）；

（6）输出 sum。

评注：上述算法中的步骤（3）、（4）、（5）组成了一个循环，在实现算法时，要反复多次执行这 3 个步骤。执行步骤（5）时，经过判断，若 i 超过规定的数值 6 则转步骤（6），输出 sum 后算法结束。此时变量 sum 的值就是所要求的结果。

用这种方法表示的算法具有较强的通用性和灵活性。如果要计算 1~100 的和，只需将步骤（5）中的 i <7 改成 i <101 即可。

【例 1-4】 请设计算法，找出[1,100]的所有质数。

分析：为了找出[1,N]（N 是任意给定的正整数）内的所有质数，只要把 1 和不超过 N 的所有合数都删去。由于不超过 N 的合数 a 必有一个不可约除数$p\left(p \leqslant \sqrt{a} \leqslant \sqrt{N}\right)$，因而，只要找出不超过$\sqrt{N}$的全部质数 p_1，p_2，…，p_s，然后依次把不超过 N 的正整数中的除了 p_1，p_2，…，p_s 以外的 p_1 的倍数，p_2 的倍数，…，p_s 的倍数全部删去，就删去了不超过 N 的全部合数，剩下的就是不超过 N 的全部质数，如图 1.1 所示。

> 该方法是古希腊的埃拉托斯色尼（Eratosthenes，约公元前 274～194 年）发明的，起初是把数写在涂腊的板上，每次要划去一个数，就在上面记以一小点。寻求质数的工作完毕后，这些小点就像一个筛子，所以就把埃拉托斯色尼发明的这种找质数的方法叫做“埃拉托斯色尼筛”，简称“筛法”。

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30

图 1.1 [1,30]埃拉托斯色尼筛示例

具体做法是：先把 N 个自然数按次序排列起来。1 不是质数，也不是合数，要划去。第二个数 2 是质数要保留下来，把 2 后面所有能被 2 整除的数都划去；2 后面第一个没划去的数是 3，把 3 留下，再把 3 后面所有能被 3 整除的数都划去；3 后面第一个没被划去的数是 5，把 5 留下，再把 5 后面所有能被 5 整除的数都划去；如此往复，直到$\sqrt{N}$为止，就会把不超过 N 的全部合数都筛掉，留下的就是不超过 N 的全部质数。筛法描述如下：

（1）将 1～100 的自然数放入 a[1]～a[100]中，并令 a[1]=0（表示划去）；

（2）找下一个非 0 最小数→newp（newp 的初始值为 2）；

（3）划去 newp 后的所有 newp 的倍数（对应位置元素值置 0）；

（4）重复（2）、（3），直至 newp>10（即$\sqrt{100}$）；

（5）输出剩余（即非零）的数。