21st CENTURY
规划教材
面向21世纪高等院校计算机系列规划教材
COMPUTER COURSES FOR UNDERGRADUATE EDUCATION

C语言程序设计

刘天印 冯运仿 主编

科学出版社
www.sciencep.com

面向21世纪高等院校计算机系列规划教材
COMPUTER COURSES FOR UNDERGRADUATE EDUCATION

C语言程序设计

刘天印　冯运仿　主编

科学出版社

北京

内 容 简 介

本书为C语言程序设计教材，以程序设计为主线，注重培养读者程序设计的思维方式和技巧，结构合理、重点突出、深入浅出、范例经典。书中每章都有内容提要、教学目标、实例分析和归纳小结，匹配有精选的练习题，以帮助读者巩固和提高。

本书由教学一线的资深教师编写，图文并茂、应用性强，适合作为各类高等院校高级语言程序设计等课程的教材，也可作为计算机程序设计培训教材或其他从事计算机程序设计的科技人员的参考书，同时还可供参加全国计算机等级考试者参考。

图书在版编目（CIP）数据

C语言程序设计/刘天印，冯运仿主编. - 北京：科学出版社，2007
（面向21世纪高等院校计算机系列规划教材）
ISBN 978-7-03-018762-8

Ⅰ.C… Ⅱ.①刘…②冯… Ⅲ.C语言－程序设计 Ⅳ.TP312

中国版本图书馆CIP数据核字（2007）第037947号

责任编辑：张颖兵／责任校对：董 丽
责任印制：高 嵘／封面设计：方葵工作室

科学出版社出版
北京东黄城根北街16号
邮政编码：100717
http://www.sciencep.com
武汉市新华印刷有限责任公司印刷
科学出版社发行 各地新华书店经销
*
2007年3月第 一 版 开本：787×1092 1/16
2007年3月第一次印刷 印张：17 1/4
印数：1—3 000 字数：417 000
定价：24.00元
（如有印装质量问题，我社负责调换）

《C语言程序设计》编委会

主　编：刘天印　冯运仿

副主编：李　芳　周松林　纪　鹏

主　审：吴定雪

前　言

C 语言既适合编写应用程序，又适合编写系统程序，既具有高级语言的优点，又具有低级语言的功能(能对硬件直接进行操作)，灵活方便，应用广泛，生成的目标程序执行效率高，可移植性强，是最具影响力的结构化程序设计语言之一。

本书共 12 章又 4 个附录，主要内容包括：

第 1 章　C 语言程序设计概述。主要向读者介绍程序设计的一般过程及方法、算法、C 语言简介，以及 C 语言的编程环境 Turbo C 2.0 和 Visual C++ 6.0。

第 2 章　C 语言的基本数据类型与表达式。重点介绍 C 语言的基本数据类型——整型、实型、字符型等，常量与变量，运算符与表达式(如算术、关系、逻辑、赋值、条件等运算符和表达式)以及数据类型转换。

第 3 章　顺序结构程序设计。主要介绍结构化程序设计方法，程序的三种基本结构——顺序结构、选择结构和循环结构，标准的输入/输出语句。

第 4 章　选择结构程序设计。主要介绍 if 语句、if…else 语句、if…else if 语句、if 语句的嵌套及 switch 语句。

第 5 章　循环结构程序设计。主要介绍 while 语句、do…while 语句、for 语句、循环的嵌套以及非结构化的 break 语句、continue 语句、goto 语句。

第 6 章　函数。重点介绍了函数的定义与调用、函数的参数传递、函数的嵌套调用与递归调用，变量的作用域，变量的存储属性(包括自动变量、寄存器变量、外部变量和静态变量)。

第 7 章　编译预处理。重点介绍了宏定义，文件包含命令和条件编译命令。

第 8 章　数组。主要介绍了一维数组、二维数组的定义、初始化及数组元素的引用，字符数组与字符串，数组作为函数参数等等。

第 9 章　指针。是 C 语言的难点之一，重点介绍指针变量的定义、运算，指针与数组，指针与字符串，指针与函数，指向指针的指针以及带参数的 main 函数等。

第 10 章　结构体、共用体、枚举及类型定义。结构体也是 C 语言的难点之一。重点介绍了结构体变量的定义、初始化和引用，结构体与数组，结构体与指针，结构体与函数，链表，共用体，枚举类型以及自定义数据类型。

第 11 章　文件。主要介绍了数据文件、文件的存取方式、文件缓冲区、文件指针，文件的操作(包括文件的打开与关闭，文件的顺序读写，文件的定位与文件的随机读写，文件操作的出错检测等)。

第 12 章　位运算。是 C 语言的特色功能。重点介绍 6 个位运算，位运算赋值运算符以及位段。

4 个附录分别是：关键字、常用字符与 ASCII 代码对照表、运算符的优先级和结合性及常用的 C 库函数。

本书由刘天印、冯运仿任主编并统稿，李芳、周松林、纪鹏任副主编，吴定雪审读了稿件。第 1 章、第 9 章、第 10 章由冯运仿编写；第 2 章、第 3 章由周松林编写；第 4 章至第 6 章由李芳编写；第 7 章、第 8 章由纪鹏编写；第 11 章、第 12 章由刘天印编写。

本书结构合理、重点突出、循序渐进、范例经典，以程序设计为主线，注重培养读者程序设计的思维方式和技巧。每章都有内容提要、教学目标、实例分析和归纳小结，每章的例子都经过 Turbo C 和 Visual C++环境的调试，每章课后的习题都经过了精心的筛选，便于读者深入理解所学内容。

本书由教学一线的资深教师编写，图文并茂、应用性强，适合作为各类高等院校高级语言程序设计等课程的教材，也可作为计算机程序设计培训教材或其他从事计算机程序设计的科技人员的参考书，同时还可供参加全国计算机等级考试者参考。

由于编者水平有限，时间仓促，书中难免有不足之处，恳请读者和专家提出宝贵意见。

编　者

2007 年 1 月

目　录

第 1 章　C 语言程序设计概述

【本章内容】

- 程序设计概述
- 算法
- C 语言简介
- C 语言编程环境

****

C 语言是在 B 语言的基础上发展起来的，它的根源可以追溯到 ALGOL 60。C 语言是一种编译型程序设计语言，它兼顾了多种高级语言的特点，并具备汇编语言的功能。C 语言程序处理功能强，运算速度快，有良好的移植性，而且可以直接实现对系统硬件及外围接口的控制，具有较强的系统处理能力。C 语言是一种结构化程序设计语言，是国际上广泛流行的计算机高级语言。C 语言程序具有完善的模块程序结构。

C 语言是数据结构、C++、操作系统等课程的前导课程，也是一门实践性很强的课程，既要掌握概念，又要动手编程，还要上机调试运行，是理工类专业的一门必修课程。同时，也是“全国计算机等级考试”二级考试的主要语种之一。

****

【教学目标】

- ☺ 了解程序设计的基本知识
- ☺ 了解 C 语言的背景
- ☺ 掌握 C 语言程序的结构，领会 C 语言程序设计的风格
- ☺ 掌握算法的基本概念与特征
- ☺ 掌握结构化程序设计的基本概念
- ☺ 熟悉 Turbo C 2.0 和 Visual C++ 6.0 编程环境

1.1　程序设计概述

1.1.1　程序与程序设计语言

1. 程序的基本概念

计算机是一种强有力的工具，能够以极高的速度对存放在主存储器里的数据进行处理，并获得期望的输出结果。原始数据的输入、处理到结果的输出，都是由人事先编写好程序，然后将程序和原始数据一同输入到计算机里，交由计算机自动完成的。所谓程序就是控制计算机完成特定功能的一组有序指令的集合，它描述了需要处理的数据以及这些数据需要执行的操作。

为解决某一个问题所编写的程序并不是唯一的，不同的用户所开发的程序也不完全相同。不同的程序有不同的效率，这涉及到程序的优化，涉及到程序所采用的数据结构以及算法等多方面的因素。

2. 程序设计语言

编写程序所使用的语言称为程序设计语言(programming language)，它是人与计算机之间进行信息交流的工具，是一组用来定义计算机程序的语法规则，是一种用来向计算机发出指令的被标准化的交流技巧。计算机语言让程序员能够准确地定义计算机所需要使用的数据，并精确地定义在不同情况下所应当采取的行动。每一种程序设计语言可以被看作是一套包含语法、词汇和含义的正式规范。这些规范通常包括数据和数据结构、指令及流程控制、引用机制和重用、设计哲学等。大多数被广泛使用或经久不衰的语言，拥有负责标准化的组织，经常创造及发布该语言的正式定义，并讨论扩展或贯彻现有的定义。程序设计语言的基本成分有：①数据成分，用于描述程序所涉及的数据；②运算成分，用以描述程序中所包含的运算；③控制成分，用以描述程序中所包含的控制；④传输成分，用以表达程序中数据的传输。

从 1946 年世界上第一台计算机诞生以来，计算机科学在不长的历史阶段中得到了迅猛发展。程序设计语言的发展从低级到高级，经历了机器语言、汇编语言、高级语言、面向对象语言的多个阶段。

1) 机器语言

计算机的硬件系统可以直接识别和执行的二进制指令(机器指令)的集合称为该种计算机的机器语言。早期的计算机程序是直接使用机器语言编写的，这种语言便于计算机直接识别，但由于机器语言与人们习惯用的语言差别太大，难学、难写、难记、难检查、难修改。而且不同机器间又不通用，给计算机的推广使用造成了很大的障碍。目前很少直接用机器语言编程。

2) 汇编语言

汇编语言将机器指令映射为一些可以被人读懂的助记符，如 ADD、MOV 等。汇编语言实际上是与机器语言相对应的语言。由于计算机只能执行机器指令，故汇编语言需要编译后才能被识别，这一过程称为汇编。同机器语言一样，汇编语言与人类的自然语言仍然相差甚远，但要比机器语言简单多了。因此，如果程序运行时间要求比较严格，程序与硬件操作联系紧密，人们还是常用汇编语言编写有关程序来解决这些问题。

3) 高级语言

1954 年出现了第一种高级语言 FORTRAN，它是一种主要用于数值计算的高级语言。高级语言屏蔽了机器的细节，提高了语言的抽象层次，程序中可以采用具有一定含义的数据命名和容易理解的执行语句，接近人类的自然语言。使用高级语言编程，一般不必了解计算机的指令系统和硬件结构，只需掌握解题方法和高级语言的语法规则，就可以编写程序。高级语言在程序设计时着眼于问题域中的过程，是一种面向过程的语言。这使得书写程序时可以联系到程序所描述的具体事物，使计算机的编程语言前进了一大步。

这个时期，随着计算机的应用日益广泛地渗透到各学科和技术领域，也发展了一系列不同风格的程序设计语言。较为著名的有 FORTRAN、COBOL、BASIC、PASCAL、Turbo C 等。

4) 面向对象的语言

20 世纪 60 年代开发的 Simula 67 被公认为面向对象语言的鼻祖，其主要用途是进行仿真

建模。70 年代和 80 年代这一时期，来自于 Simula 和其他早期的原型语言中的面向对象的概念在 SmallTalk 语言中得到了完整的体现。面向对象的编程语言与以往各种编程语言的根本不同点在于，它设计的出发点就是为了能更直接地描述客观世界中存在的事物(即对象)以及它们之间的关系。面向对象的语言的出现改变了编程者的思维方式，使设计程序的出发点由着眼于问题域中的过程转向着眼于问题域中对象及其相互关系，这种转变更加符合人们对客观事物的认识。因此，面向对象的方法更接近自然语言，是人们对于客观事物更高层次的抽象。

面向对象的语言已形成几大类别：一类是纯面向对象的语言，如 SmallTalk 和 Eiffel；一类是混合型的面向对象语言，如 C++ 和 Objective C；还有一类是与人工智能语言结合形成的，如 LOOPS、Flavors、CLOS 以及适合网络应用的 JAVA 语言等。

综上所述，每一种语言都有它的优势和劣势。对于不同的问题，要根据实际情况来选择程序设计语言，以便更高效、更优质地解决相关问题。

1.1.2　程序设计的一般过程

程序设计实际上就是选择使用一种程序设计语言编制程序的过程，具体讲就是数据结构与算法的统一过程。

进行程序设计通常需要经过的处理步骤如图 1.1 所示。首先要明确需要解决的问题是什么，即提出问题；其次要分析问题中涉及了哪些数据，如何在计算机中进行表示，即描述数据结构；同时还要将复杂的问题分解为计算机可以完成的若干操作步骤，即确定算法；然后用选定的某种程序设计语言描述数据结构，并根据算法编写程序；编好的源程序输入计算机后，往往需要进行反复调试，修正其中的语法错误和逻辑错误，直至得到正确的运算结果。

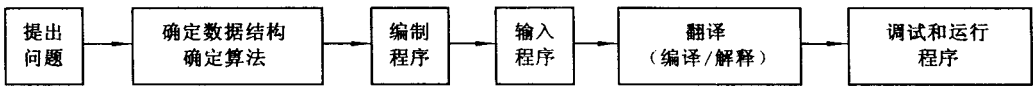

图 1.1　程序设计的一般过程

从上述过程中可以看出，使用计算机进行计算必须先进行程序设计，而程序设计的关键是确定算法。实际上只要算法正确，用哪一种高级语言实现并不重要。

注意：计算机不能直接识别和执行高级语言源程序，必须经过翻译，才能将其转换成机器语言程序执行。翻译的方法有两种：一种是编译方式，即使用一种称为编译程序的软件，将高级语言源程序整个翻译成机器语言目标程序，再经过连接生成可执行的程序；一种是解释方式，即通过一种称为解释程序的软件对源程序逐句执行，也就是边解释边执行，不生成目标程序。

1.1.3　程序设计方法

计算机程序设计方法的发展大致经过了面向机器的方法、面向过程的方法和面向对象的方法等几个阶段。

1. 面向机器的方法

面向机器的方法使用机器语言或汇编语言。设计一个程序基本过程为：写出算法，验证算法，转换成机器语言，然后调试代码，直到产生希望的结果。

这种方法基本上是“软件作坊”式的，与人类的自然语言之间存在着巨大的鸿沟，程序员需要考虑大量的机器细节，软件开发的难度大、周期长，开发出的软件功能却很简单，界面也不友好。这种局面严重影响了计算机的普及与应用。

2. 面向过程的方法

高级语言的蓬勃兴起，使得编译和形式语言理论日趋完善，但就整个程序设计方法而言，并无实质性的改进。

自 20 世纪 60 年代末到 70 年代初，出现了大型软件系统，如操作系统、数据库，这给程序设计带来了新的问题。大型系统的研制需要花费大量的资金和人力，可是研制出来的产品却是可靠性差，错误多，且不易维护和修改，在软件开发和维护过程中遇到一系列严重问题，当时人们称这种现象为“软件危机”。为此人们开始重新审视程序设计中的一些最基本的问题。例如，程序的基本组成是什么，应该用什么样的方法来设计程序，如何保证程序设计正确，程序设计的主要方法和技术应如何规范等等。

1969 年，E. W. Dijkstra 首先提出了结构化程序设计的思想，提出了从程序结构和风格上来研究程序设计。用结构化程序设计的方法编写出来的程序不仅结构良好，易读易写，而且其正确性易于证明。

20 世纪 70 年代末 Niklaus Wirth 又提出了“算法+数据结构=程序设计”的结构化程序设计方法，他将软件系统分解为若干可单独命名和编址的功能模块，分别编写各功能模块的代码，最后将各模块联结、组合成相应结构的软件系统。模块化使软件能够有效地被管理和维护，能够有效地分解和处理复杂问题。到 80 年代，模块化程序设计方法普遍被人们所接受。

3. 面向对象的方法

20 世纪 80 年代之后，人们又提出了面向对象的程序设计方法。面向对象的方法是把软件系统分解成为相互协作而又彼此独立的对象的集合，每个对象有自己的数据、操作和功能，把对象看作由数据及可以施加在这些数据上的操作所构成的统一体。对象之间通过“消息”进行通信。这种方法直接对问题域中的客观事物建造分析模型中的对象，使对象的描述与客观事物一致。保持问题域中的单个事物及事物之间的关系的原貌。

面向对象的编程语言可以直接描述问题域中的对象及其相互关系。它将客观事物看作具有属性和行为的对象，通过抽象找出同一类对象的共同属性(静态特征)和行为(动态特征)形成类，对象之间通过“消息”进行通信。通过类的继承与多态可以方便地实现代码重用，便于实现功能的扩充、修改，增加或删除，从而降低软件的调试、维护难度。这一方法特别适合于需要多人合作的大型软件的开发。

面向对象的方法=对象+类+继承+消息通信。面向对象方法的核心是，将问题分解为对象的若干类，并寻找诸类之间的关系和相互作用。其主要特点是将数据和应用在数据上的操作封装在一起。通过封装、继承、多态、消息等机制实现类(对象)之间的联系与相互作用。

1.2 算 法

1.2.1 算法的概念

一个程序应包括对数据的描述和对数据处理的描述。对数据的描述，即数据结构，是计算机学科的核心课程之一，有许多专门著作论述，本课程就不再赘述；对数据处理的描述，即动作的描述，也就是算法，是为解决一个特定问题而采取的方法和确定的有限的步骤。

本书介绍的算法只限于计算机算法，即计算机能够执行的算法。计算机算法分为数值运算算法和非数值运算算法。解决数值计算问题的算法叫做数值算法，科学和工程计算方面的算法都属于数值算法，如求解数值积分、求解线性方程组、求解微分方程等。解决非数值问题的算法叫做非数值算法，数据处理方面的算法都属于非数值算法。非数值运算包括的面很广，最常见的是应用于事务管理领域，如图书检索、人事管理等。由于数值运算有现成的模型，可以运用数值分析方法，因此对数值运算的算法研究比较深入，算法比较成熟，对各种数值运算都有比较成熟的算法可供选用。而非数值运算的种类繁多，要求各异，难以规范化，因此只对一些典型的非数值运算算法做比较深入的研究。其他的非数值运算问题，往往要对特定的问题重新设计专门算法。

在计算机领域，一个算法实质上是针对所处理问题的需要，在数据的逻辑结构和物理结构的基础上，施加的一种运算。由于数据的逻辑结构和物理结构不是唯一的，在很大程度上可以由用户自行选择和设计，所以处理同一个问题的算法也不是唯一的。另外，即使对于相同的逻辑结构和物理结构而言，其算法的设计思想和技巧不同，编写出的算法也大不相同。例如，求 1+2+3+…+100，有人可能先用 1+2，再加 3，一直加到 100；而有的人可能采用 100+(1+99)+…+(49+51)+50=5050。当然还可以有其他的一些方法。有的方法只需进行很少的步骤，而有些方法则需要较多的步骤。一般说，人们都希望采用简单的和运算步骤少的方法。因此，为了有效地进行计算，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。

一个算法应该具有以下特点：

(1) 有穷性。一个算法应该在有限的操作步骤内完成。任何一个问题的解决不论其采取什么样的算法，其终归是要把问题解决好。如果一种算法的执行时间是无限的，或在期望的时间内没有完成，那么这种算法就是无用和徒劳的，就不能称其为算法。

(2) 确定性。算法中的每一个步骤都应当是确定的，不应该是含糊或模棱两可的，即算法中的每一个步骤应当是十分明确无误的，不能有歧义。

(3) 有 0 个或多个输入。所谓输入是指在执行算法时需要从外界取得必要的信息。例如，判断一个数据是否是素数，此时只有一个输入，而计算两个数的最小公倍数时，输入应该是两个。一个算法也可以没有输入。例如求已知数据的累加和，显然不需要再输入任何数据就可以完成该算法。

(4) 有一个或多个输出。算法的目的是为了求解，该“解”就是输出。例如求最小公倍数的算法，最后打印出的公倍数就是输出。但算法的输出不一定就是计算机打印输出，一个算法得到的结果就是算法的输出。没有输出的算法是没有意义的。

(5) 有效性。一个算法必须遵循特定条件下的解题规则，组成它的每一个操作都应该是