高等学校计算机专业规划教材

C语言程序设计

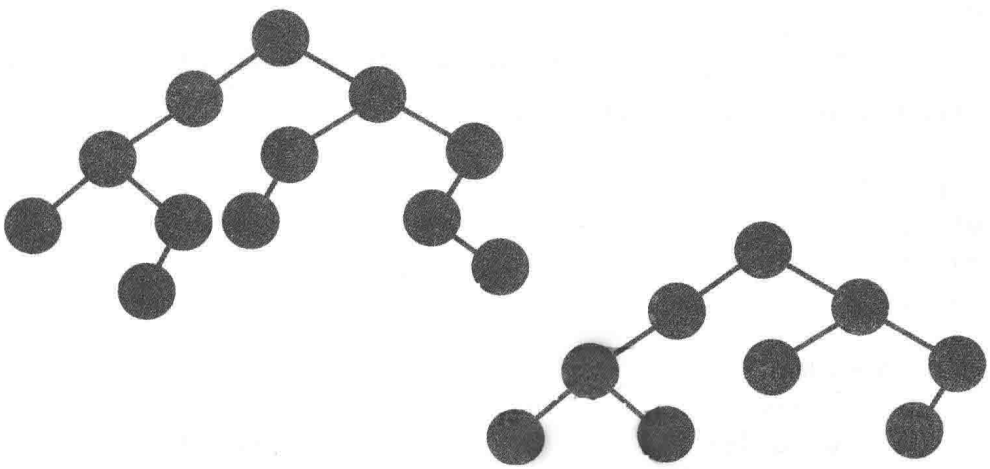
李忠月 励龙昌 主 编
虞铭财 黄海隆 副主编

清华大学出版社

高等学校计算机专业规划教材

C语言程序设计

李忠月 励龙昌 主 编
虞铭财 黄海隆 副主编



清华大学出版社
北 京

内 容 简 介

本书采用“实例导入-问题提出-解释与应用”这一基本叙述模式，引入生动有趣的案例，从情境中提出问题，建立数学模型，获得解决，最后再应用。全书共分11章，包括概述、分支结构、循环结构、函数数组、指针、结构等内容。

本书在结构设计上，从有利于学生学习的角度出发选择、组织和呈现教学内容。首先，在书的安排顺序上，先安排函数，然后是数组和指针，这样便于学生早接触函数，早使用函数，有利于学生后续课程的学习；其次，强调实践，而不拘泥于基础知识，通过实践掌握基础知识，重点在程序设计能力的培养；再次，本教材设计了一些专题，如迭代算法、素数判定等，总结了某一类问题的解决方法，又让学生体验到程序设计的实用性，激发了学生的学习兴趣。

本书可以作为各类大专院校、等级考试与各类培训的教学用书，也可作为C语言程序设计的自学用书。

图书在版编目(CIP)数据

C语言程序设计/李忠月，励龙昌主编. —北京：清华大学出版社，2014
高等学校计算机专业规划教材
ISBN 978-7-302-37452-7

Ⅰ. ①C…　Ⅱ. ①李… ②励…　Ⅲ. ①C语言－程序设计－高等学校－教材　Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2014)第170680号

责任编辑：龙启铭
封面设计：常雪影
责任校对：焦丽丽
责任印制：李红英

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦A座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn,010-62795954

印 装 者：北京国马印刷厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：24.5　　**字　　数**：564千字
版　　次：2014年9月第1版　　**印　　次**：2014年9月第1次印刷
印　　数：1～2000
定　　价：39.00元

产品编号：060950-01

前言

“程序设计”是高校理工科重要的计算机基础课程，该课程以培养学生掌握程序设计的思想和方法为目标，以培养学生的实践能力和创新能力为重点。C语言是得到广泛使用的程序设计语言之一，它既具备高级语言的特性，又具有直接操纵计算机硬件的能力，并以其良好的程序结构和便于移植的特性而拥有大量的使用者。目前，许多高校都把C语言列为首门要学习的程序设计语言。

虽然目前有关C语言的教材很多，但大多比较注重C语言知识的学习，不利于培养学生的程序设计能力和语言应用能力。本书以程序设计为主线，从应用出发，通过案例和问题引入相关的语法知识，重点讲解程序设计的思想和方法，并始终贯彻整本书，避免机械式地记忆语法知识，坚持通过写程序去掌握C语言知识的理念。

在教材的结构设计上，强调学以致用，使学生从一接触C语言，就开始练习编程。全书共11章，主要包括3方面的内容：基本内容、常用算法和程序设计风格。为了提高读者的学习兴趣，大多是先导入实例而后介绍相关的语言知识。第1章简单介绍一些背景知识和利用计算机解决问题的步骤，然后从实例出发，简要介绍C语言的核心部分，使学生对C语言有一个总体的了解，并学习编写简单的程序，培养学习兴趣；第2章介绍基本的数据类型和常用运算符；第3章和第4章分别介绍分支结构和循环结构程序设计的思路和方法，从第3章开始，逐步深入介绍程序设计的思想和方法，说明如何应用语言解决问题；第5章介绍基本的输入输出处理；第6章介绍函数的基本知识及基本用法；第7章介绍一维数组、二维数组和字符数组的知识和应用；第8章介绍指针的基本概念及应用；第9章介绍结构的基本知识及应用；第10章介绍位运算及应用；第11章介绍文件的概念、文件的基本操作及应用。

本书具有如下特色。

(1) 注重知识内容的实用性和综合性。结合应用型本科学生的特点，注重知识内容的实用性和综合性，删减以往类似教材中较刻板的理论知识点，将更多的篇幅放在程序设计方法、程序设计技能以及程序设计过程的阐述上。

(2) 设计了一些专题。本书安排了正整数的拆分、最大公约数、素数、进制转换等专题。这些专题既总结了某一类问题的解决方法，又让学生体验到程序设计的实用性，激发了学生的学习兴趣。

(3) 本书图文并茂。西方有句谚语："A picture is worth a thousand words"(一图值千言),意思是用上千个字描述不明白的东西,很可能一张图就能解释清楚。本书基本上做到对难理解部分的介绍都有相关的图示,有的内容通过多图逐步分解剖析。

(4) 本书在提供丰富、有趣的经典实例时,还精心设计了一个相对完整的"学生成绩管理"应用实例贯穿于整本书,从最简单的单个学生成绩处理开始,到用循环语句、数组处理多个学生成绩信息,再到用更有聚合力的结构来组织学生成绩信息,最终能将这些处理信息永久性地存储到文件中为止,完全贯彻实用、实践和工程应用的理念。通过这个实例的学习,让学生对C语言程序设计有一个更全面的认知,能够综合运用所学知识去解决较为实际的问题。

因编者水平有限,对书中存在的疏漏、谬误之处,敬请读者批评指正。

编者

2014年6月

目录

第 1 章 概　述

本章要点：

- 计算机程序设计语言的发展过程；
- 用程序设计语言编写程序的步骤；
- 结构化程序设计的方法；
- 算法的定义、算法的特性以及算法的表示；
- C 语言的特点；
- C 语言程序的基本框架；
- 进行 C 语言程序设计所需要的一些基本元素。

现代计算机是一种通用机器，具有很大的潜力，但必须对其进行编程才能发挥出这些潜力。给计算机编程就是给它一组指令（即一个程序），这组指令详细地指定解决问题的每一个必要步骤。

程序需要用某种语言来描述，例如，用算盘进行计算时，程序是用口诀来描述的，而现代计算机的程序则是用计算机程序设计语言来描述的。

1.1　计算机程序设计语言

从计算机诞生至今，计算机程序设计语言也在伴随着计算机技术的进步不断发展，种类非常多，总的来说可以分成机器语言、汇编语言、高级语言三大类。

1. 机器语言

CPU 指令系统（也称 CPU 的机器语言）是 CPU 可以识别的一组由 0 和 1 序列所构成的指令码。

用机器语言（Machine Language）编写程序，就是从所使用的 CPU 指令系统中挑选合适的指令，组成一个指令序列。这种程序虽然可以被机器直接理解和执行，但由于其不够直观、难记、难认、难理解、不易查错而只能被少数专业人员所掌握，而且编写程序的效率很低，质量难以保证。这种繁重的手工编写方式与高速、自动工作的计算机极不相称。这种方式仅用于计算机出现的初期编程，现在已经不再使用。

2. 汇编语言

为了降低编写程序过程中的劳动强度，20 世纪 50 年代中期人们开始用助记符（Memoni）代替操作码，用地址符号（Symbol）或标号（Label）代替地址码，这样用符号代替

机器语言的二进制码，就把机器语言变成了汇编语言(Assembly Language)，因此汇编语言也称为符号语言。

机器不能直接识别，用汇编语言编写的程序，需要由一种程序将汇编语言翻译成机器语言，这种起翻译作用的程序称为汇编程序。汇编程序是系统软件中语言处理系统软件。汇编语言编译器把汇编程序翻译成机器语言的过程称为汇编。

汇编语言比机器语言易于读写、调试和修改，同时具有机器语言的全部优点。但在编写复杂程序时，相对高级语言来说，代码量较大，而且汇编语言依赖于具体的处理器体系结构，不能通用，因此不能在不同处理器体系结构之间直接移植。

汇编语言和机器语言都是面向机器的程序设计语言，一般称为低级语言。

3. 高级语言

由于汇编语言依赖于硬件体系结构，且助记符量大、难记，于是人们又发明了更加易用的高级语言(High-level Language)。高级语言主要是相对于汇编语言而言，它并不是特指某一种具体的语言，而是包括了很多编程语言，如 C、C++、Delphi、Java 等。

高级语言所编制的程序不能直接被计算机识别，必须经过转换才能被执行。按转换方式不同，可将它们分为解释型和编译型这两大类。解释型是指应用程序源代码一边由相应语言的解释器翻译成目标代码(机器语言)，一边执行，因此效率比较低，而且不能生成可独立执行的可执行文件，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。编译型是指将应用程序源代码翻译成目标代码(机器语言)，因此其目标程序可以脱离其语言环境独立执行，使用比较方便、效率较高，但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(*.obj)才能执行。由于只有目标文件而没有源代码，修改很不方便。现在大多数的编程语言都是编译型的。

高级语言与计算机的硬件结构及指令系统无关，它有更强的表达能力，可方便地表示数据的运算和程序的控制结构，能更好地描述各种算法，而且容易学习和掌握。但高级语言编译生成的程序代码一般比用汇编程序语言设计的程序代码要更长，执行的速度也更慢，所以汇编语言适合编写一些对速度和代码长度要求高的程序，以及一些要直接控制硬件的程序。

1.2 用程序设计语言编写程序的步骤

1.2.1 编码

用计算机解决问题包括两个步骤：

① 应该构造出一个算法或在解决该问题的已有算法中选择一个，这个过程称为算法设计；

② 用程序设计语言将该算法表达为程序，这个过程称为编码。

1.2.2 编译

为了使用高级语言编写的程序能够在不同的计算机系统上运行，首先必须将程序翻

译成运行程序的计算机所特有的低级机器语言。在高级语言和机器语言之间执行这种翻译任务的程序称为编译器。

编译器将源文件翻译成目标文件，其中包含适用于特定计算机系统的实际指令，这个目标文件和其他目标文件可组成在系统上运行的可执行文件。这些所谓的其他目标文件常常是一些称为库的预定义的目标文件，库中含有程序所要求的不同操作的机器指令。将所有独立的目标文件组合成一个可执行文件的过程称为连接。高级语言程序的执行过程如图 1-1 所示。

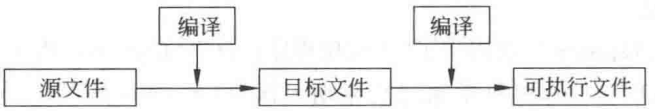


图 1-1 高级语言程序的执行过程

1.2.3 调试

程序设计语言有自己的语法，它决定如何将一个程序的元素组合在一起。编译一个程序时，编译器首先检查程序的语法是否正确，由于违反语法规则而导致的错误称为语法错误。当从编译器得到一个语法错误的消息时，必须返回程序并改正错误。语法错误比较容易改正。

通常，程序运行失败往往不是由于语法错误，而是由于合乎语法的程序有一个逻辑上的错误，程序员称这种错误为逻辑错误(即 bug)。找出并改正这种逻辑错误的过程称为调试(debugging)，它是程序设计过程中重要的一环。

所有的程序员不仅会犯逻辑错误，而且有时还会制造一系列逻辑错误。优秀程序员的优秀之处并不在于他们能够避免逻辑错误，而是他们能努力将存在于完成的代码中的逻辑错误数量减到最少。

1.2.4 维护

软件开发的一个特殊方面是程序需要维护。软件开发完成交付用户使用后，就进入了软件的运行和维护阶段。

软件需要维护主要有两个原因：首先，即使经过大量测试，并在相关领域使用多年，源代码中依然可能存在逻辑错误；其次，当出现一些不常见的情况或发生之前未预料到的情况时，之前隐藏的逻辑错误就会使程序运行失败。

软件维护工作处于软件生命期的最后阶段，维护阶段是软件生存期中最长的一个阶段。软件维护很困难，尤其是对大型、复杂系统的维护，更加困难和复杂。

软件维护的困难是由于软件需求分析和开发方法的缺陷。这种困难表现在如下几个方面：

① 读懂别人的程序比较困难；

② 文档的不一致性；

③ 软件开发和软件维护在人员和时间上的差异。

在软件维护阶段所花费的人力、物力最多，其花费占整个软件生命期花费的约60%～70%。因此，应该充分认识到维护工作的重要性和迫切性，提高软件的可维护性，减少维护的工作量和费用，延长已经开发软件的生命期，以发挥其应有的效益。

1.3 结构化程序设计方法

程序设计方法的发展可以划分为以下三个阶段：早期的程序设计、结构化程序设计、面向对象的程序设计。

1969年，E. W. Dijkstra首次提出了“结构化程序设计”(Structured Programming)的概念。

1971年4月，瑞士计算机科学家尼克莱斯·沃思(Niklaus E. Wirth)在*Communications of ACM*上发表了*Program Development by Stepwise Refinement*一文，提出了“通过逐步求精方式开发程序”的思想。

结构化程序设计的基本思想是，对大型的程序设计，使用一些基本的结构来设计程序，无论多复杂的程序，都可以使用这些基本结构按一定的顺序组合起来。

这些基本结构是指按指令的顺序依次执行的顺序结构、根据判别条件有选择地改变执行流程的分支结构、有条件地重复执行某个程序块的循环结构。

这三种基本结构有以下共同点：

① 只有一个入口，不得从结构外随意转入结构中的某点；

② 只有一个出口，不得从结构内的某个位置随意转出；

③ 结构中的每一部分都有机会被执行到，即没有“死语句”；

④ 结构内不存在“死循环”。由这些基本结构组成的程序就避免了任意转移、阅读起来需要来回寻找的问题。

结构化设计思想的核心不是要求一步就编制成可执行的程序，而是分若干步进行，逐步求精。它的具体内容如下：

(1) 要求把程序的结构规定为顺序、选择和循环3种基本结构，并提出了自顶向下、逐步求精、模块化程序设计等原则。

(2) 结构化程序设计是把模块分割方法作为对大型系统进行分析的手段，使其最终转化为3种基本结构，其目的是解决由许多人共同开发大型软件时，如何高效率地完成可靠系统的问题。

(3) 程序的可读性好、可维护性好成为评价程序质量的首要条件。

它的缺点是：程序和数据结构松散地耦合在一起。解决此问题的方法就是采用面向对象的程序设计(Object oriented programming，OOP)方法。

C程序设计语言就是结构化程序设计语言。

1.4 算　　法

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，即为数据结构。在C语言中，系统提供的数据结构是以数据类型的形式出现的。对数据处理的描述，即

为计算机算法。

算法是规则的有限集合，是为解决特定问题而规定的一系列操作，是有限的。对于同一个问题可以有不同的解题方法和步骤，也就是有不同的算法。算法有优劣，一般而言，应当选择简单的、运算步骤少的，即运算快、内存开销小的算法，也就是要考虑算法的时空效率。

程序设计是一门艺术，主要体现在结构设计和算法设计上，结构设计艺术好比是程序的肉体，算法设计是好比是程序的灵魂。著名的计算机科学家沃思提出一个公式：

数据结构＋算法＝程序

实际上，一个程序除了数据结构和算法外，还必须使用一种计算机语言，并采用结构化方法来表示。

1.4.1 算法的特性

一个算法应该具有以下特性。

(1) 有穷性：算法必须在执行有穷步之后结束，而每一步都必须在有穷时间内完成。

(2) 确定性：算法中的每个步骤都必须是确定的，不能有二义性。

(3) 可行性：一个算法必须是可行的，即算法中每一操作都能通过已知的一组基本操作来实现。

(4) 输入：一个算法可以有零个或多个输入。有的算法不需要从外界输入数据，如计算 1＋2＋…＋100；而有的算法则需要输入数据，如计算 1＋2＋…＋n，执行时需要从键盘输入 n 的值后才能计算。

(5) 输出：一个算法有一个或多个输出。算法的实现是以得到计算结果为目的的，没有任何输出的算法是没有任何意义的。

1.4.2 算法的描述

1966 年，Bohra 和 Jacopini 提出了顺序结构、分支结构和循环结构这三种基本结构，用这三种基本结构作为描述一个良好算法的基本单元。已经证明，由这三种基本结构顺序组成的算法结构可以解决任何复杂问题。由基本结构组成的算法属于“结构化”算法。

为了描述一个算法，可以用不同的方法。常用的方法有：流程图、N-S 图、伪代码、计算机语言等。

1. 用流程图描述算法

流程图是用一些约定的几何图形来描述算法。用某种框图表示某种操作，用箭头表示算法流程。流程图是程序的一种比较直观的表示形式，美国标准化协会 ANSI 规定了一些常用的流程图符号，已为世界各国程序工作者普遍采用，如图 1-2 所示。

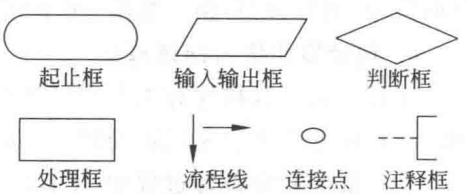


图 1-2 常用的流程图符号

三种基本结构的流程图表示如图 1-3 所示。