高等院校信息技术应用型规划教材

C语言程序设计

江义火 姜德森 苏荣聪 编著

清华大学出版社

高等院校信息技术应用型规划教材

C语言程序设计

江义火 姜德森 苏荣聪 编著

清华大学出版社
北 京

内 容 简 介

本书从程序设计的基本概念入手，对C语言的基本数据元素、运算符与表达式、流程控制语句、构造数据类型、函数、指针等内容进行由浅入深的讲解。各章内容从示例入手，尽可能将概念、知识点与例题结合起来，每章结尾均对该章内容进行小结，章末附有不同类型的习题。除第1章外，每章还设置有数量不等的实验内容。本书所有的例题都在Turbo C和Visual C++ 6.0环境下调试通过。本书配有丰富的教学资源，内容包括各章例题源程序、课程教案、习题答案和实验指导，读者可从http://www.tup.com.cn下载。

本书可作为高等院校C语言程序设计课程的教材，也可作为各类培训班的培训教材，还可作为相关技术人员的技术参考书。

图书在版编目(CIP)数据

C语言程序设计/江义火，姜德森，苏荣聪编著. —北京：清华大学出版社，2012.1
ISBN 978-7-302-27844-3

Ⅰ. ①C…　Ⅱ. 江…　②姜…　③苏…　Ⅲ. ①C语言—程序设计　Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2011)第279789号

责任编辑：孟毅新
责任校对：李　梅
责任印制：李红英
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：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：北京鑫海金澳胶印有限公司
经　　销：全国新华书店
开　　本：185×260　**印　张**：20.5　**字　数**：496千字
版　　次：2012年1月第1版　　**印　　次**：2012年1月第1次印刷
印　　数：1～4000
定　　价：39.00元

产品编号：045097-01

前　言

C语言是常用的程序设计语言之一，具有功能丰富、语句简洁、语法灵活、数据结构多样、能对硬件进行操作、目标程序效率高、可移植性好等诸多优点，适合用来编写系统软件和应用软件。

C语言程序设计是我国大部分高等院校都开设的专业基础课程，是计算机相关专业程序设计入门非常重要的必修课程。在编写本书的过程中，作者结合多年在高等院校从事C语言程序设计教学的经验，理论结合实际，力求通俗易懂。本书在体系结构安排上，根据教学目的和要求，各章以示例入手，尽可能将概念、知识点与例题结合起来，每章结尾均对该章内容进行小结，章末附有不同类型的习题。除第1章外，每章还设置有数量不等的实验内容。全书共10章，从程序设计的基本概念入手，对C语言的基本数据元素、运算符与表达式、流程控制语句、构造数据类型、函数、指针等内容的主要方面进行由浅入深的讲解。本书的特点是将基本概念和解题思路融入例题当中，借助“说明”和“注意”等教学提示，帮助读者理解教学内容；所选例题比较典型，针对性强，与所讲知识点联系紧密，突出实用性和易学性；学生完成每章的习题后都能加深理解和进一步巩固课堂所学知识；通过各章设置的实验进行实践，使学生边学边练，融会贯通、举一反三，逐步深入扩展编程训练，提高程序设计能力。

本书所有的例题均在Turbo C和Visual C++ 6.0环境下调试通过，为方便教师教学和学生学习，本书提供了丰富的实例和数字教学资源，内容包括各章例题源程序、课程教案和实验指导书，读者可从http://www.tup.com.cn下载。

本书第1、6、7、10章由江义火编写，第3、8、9章由苏荣聪编写，第2、4、5章由姜德森教授编写。全书的审核与统稿工作由姜德森完成。

在编写本书的过程中参考了相关文献，在此向这些文献的作者深表感谢。同时对关心和支持本书编写的领导和同志表示由衷的谢意。

由于作者水平有限，书中难免有不足之处，恳请专家和读者批评指正。我们的信箱是CLanPro@126.com。

编　者

2011年10月

目　录

第1章　程序设计概述

教学目标、要求

通过本章的学习，要求了解程序、程序设计语言、语言处理程序、算法及C语言的基本概念；熟悉设计程序的基本原则、算法的表示方法、结构化程序设计方法、C语言的特点；掌握自然语言、程序流程图、伪代码表示算法的方法、结构化程序设计及C语言程序的结构。

教学用时、内容

本章教学共需6学时，其中理论教学4学时，实践教学2学时。教学主要内容如下：

- 程序和程序设计语言
- 算法
  - 算法的概念
  - 算法的表示方法
    - 自然语言
    - 程序流程图
    - N-S流程图
    - 伪代码
    - 计算机语言
- 结构化程序设计
  - 基本控制结构
  - 设计方法
- C语言概述及开发工具
  - C语言产生与发展
  - C语言的特点
  - C语言的程序结构
  - C语言的开发工具简介

教学重点、难点

重点：(1) 设计程序的基本原则；
(2) 算法的表示方法；
(3) 结构化程序设计的基本结构；
(4) C语言的程序结构。

难点：算法的表示方法。

1.1　程序和程序设计语言

电子计算机的诞生是20世纪人类最伟大的发明之一，它对人类影响极其深远。自计算机问世以来，随着计算机硬件和软件不断升级换代以及计算机应用领域的迅速扩大，计算机程序设计语言也有了很大的发展。

1.1.1　程序与程序设计的概念

计算机程序(Program)是按程序设计的计算机指令的集合，它告诉计算机如何完成一

个具体的任务。在《计算机软件保护条例》中，计算机程序是指为了得到某种结果，而由计算机等具有信息处理能力的装置执行的代码化指令序列，或者可被自动转换成代码化指令序列的符号化指令序列或者符号化语句序列。

程序设计(Programming)是指设计、编制、调试程序的过程，即根据要解决的问题，使用某种程序设计语言，设计出能够完成这一任务的计算机指令序列。一个计算机程序应包括以下两部分。

(1) 对数据的描述。在程序中要指定数据的类型和数据的组织形式，即数据结构(Data Structure)。

(2) 对操作的描述。即操作步骤，也就是算法(Algorithm)。

Niklaus Wirth 提出的公式如下：

数据结构＋算法＝程序

本书编者认为：

程序＝算法＋数据结构＋程序设计方法＋语言工具和环境

这 4 个方面知识是程序设计人员必备的。

程序设计可根据不同的标准进行分类。

(1) 按照结构性质，有结构化程序设计与非结构化程序设计之分。前者是指具有结构性的程序设计方法与过程，它具有由基本结构构成复杂结构的层次性；后者反之。

(2) 按照用户的要求，有过程式程序设计与非过程式程序设计之分。前者是指使用过程式程序设计语言的程序设计；后者指非过程式程序设计语言的程序设计。

(3) 按照程序设计的成分性质，有顺序程序设计、并发程序设计、并行程序设计、分布式程序设计之分。

(4) 按照程序设计风格，有逻辑式程序设计、函数式程序设计、对象式程序设计之分。

程序设计规范是进行程序设计的具体规定。程序设计是软件开发工作的重要部分，而软件开发是工程性的工作，所以要有规范。程序设计语言影响程序设计的功效以及软件的可靠性、易读性和易维护性。专用程序为软件人员提供合适的环境，便于进行程序设计工作。

1.1.2 程序设计语言

程序设计语言是人与计算机进行交流的一种形式语言，是人利用计算机分析问题、解决问题的一个基本工具。人类的自然语言不能被计算机所直接理解，但作为人与计算机沟通的计算机程序设计语言同人类自然语言一样，也是由字、词、句子和语法等构成的指令系统。

程序设计语言，通常简称为编程语言，是一组用来定义计算机程序的语法规则。它是一种被标准化的交流技巧，用来向计算机发出指令。一种计算机语言使程序员能够准确地定义计算机需要使用的数据，并精确地定义在不同情况下应当采取的行动。

计算机程序设计语言的发展经历从低级到高级，从具体到抽象，直到可以用自然语言来描述。其种类非常多，总的来说可以分成机器语言、汇编语言和高级语言三大类。

(1) 机器语言是直接用二进制代码指令表达的计算机语言，指令是用 0 和 1 组成的一串代码，如某种计算机的指令为 1011011000000000，它表示使计算机进行一次加法操作；而指令 1011010100000000 则表示进行一次减法操作。它们的前八位表示操作码，而后八位表

示地址码。机器语言或称为二进制代码语言，计算机可以直接识别，不需要进行任何翻译。每台机器的指令，其格式和代码所代表的含义都是硬性规定的，由计算机硬件直接实现，故称为面向机器的语言，也称为机器语言。

机器语言是第一代计算机语言，它对不同型号的计算机来说一般是不同的。机器语言的优点是计算机不仅可直接识别、执行，而且运行速度快。但是，其可读性、可移植性和重用性都很差，使用机器语言编写程序非常困难，它不仅难学、难记忆、难理解、难维护，而且容易出错。

(2) 汇编语言是用一些“助记符”来表示二进制数机器指令，例如用英文缩写 ADD 表示加法运算，SUB 表示减法运算，用一些其他形式的数字和符号表示数值、存储单元地址等。汇编语言是由于使用机器语言编写程序非常困难而出现的，它的指令与机器指令基本上是一一对应的，它便于记忆和使用，同时具有机器语言的全部优点，因而用其编写程序比较容易读写、调试和修改。但在编写复杂程序时，相对高级语言，其代码量较大，而且汇编语言依赖于具体的处理器体系结构，不能通用，因此不能直接在不同处理器体系结构之间移植。

使用汇编语言编写的程序，机器不能直接识别，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叫汇编程序，汇编程序是系统软件中的一种语言处理程序。汇编程序把汇编语言编写的程序翻译成机器语言的过程称为汇编。

(3) 由于汇编语言依赖于硬件体系，且助记符量大难记，于是人们又发明了更加易用的所谓“高级语言”。从 20 世纪 50 年代中期陆续产生了许多高级语言，这些高级语言不依赖具体机器，用接近于数学语言和自然语言的方式来描述解决问题的方法和步骤。高级语言独立于机器，编程者在不了解机器内部构造和特点的情况下，也可以编写出实用的程序；由于高级语言具有通用性，用高级语言编写的程序，可以在不同类型的计算机上运行，可移植性好。高级语言由于易学、易掌握，且设计出来的程序可读性好、可维护性强、可靠性高，因此迅速得以推广和使用。

尽管高级语言有诸多优点，但是用高级语言所编制的程序(称为源程序)不能被计算机直接识别，必须经过转换才能被执行，这种转换可分为两类：编译方式和解释方式。所谓编译方式是指将源程序代码先编译成目标代码(即“翻译”成用机器语言表示的“目标程序”)，然后再执行目标程序。由于目标程序可以脱离其高级语言环境而独立执行，因此使用比较方便、效率较高。但是，应用一旦改变，必须先修改源代码，再重新编译生成新的目标程序才能执行(当只有目标程序而没有源代码时，则修改很不方便)。现在大多数编程语言都是编译型的语言，例如 Visual C++、Visual FoxPro、Delphi、FORTRAN 等。解释方式类似于日常生活中的“同声翻译”，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，而且不能生成独立的可执行的目标程序文件。但这种方式比较灵活，可以动态地调整、修改应用程序。BASIC 语言属于解释型的语言。

目前在各领域经常使用的高级语言主要有以下几种。

(1) FORTRAN 语言是第一个出现的高级语言，它是 1954 年美国 IBM 的 IT 成果。开始是为解决数学问题和科学计算而提出的，由于 FORTRAN 本身具有标准化程度高、便于程序互换、较易优化、计算速度快等特点，目前仍在使用。

(2) BASIC 语言是由 Dartmouth 学院 John G. Kemeny 与 Thomas E. Kurtz 两位教授于 20 世纪 60 年代中期所创。现今 BASIC 已有多个版本，其 Windows 环境下的 Visual BASIC 是

一个功能强大的可视化软件开发工具。

(3) Pascal 语言是由瑞士 Niklaus Wirth 教授于 20 世纪 60 年代末设计的。Pascal 语言可以方便地用于描述各种算法与数据结构，尤其对程序设计初学者，Pascal 语言有益于培养良好的程序设计风格和习惯。

(4) C 语言的原型 ALGOL 60 语言。它既具有高级语言的特点，又具有汇编语言的特点。它可以作为系统程序设计语言，编写系统软件，也可以作为应用程序设计语言，编写不依赖计算机硬件的应用程序。因此，它的应用范围广泛。

(5) C++语言是一种优秀的面向对象程序设计语言，它在 C 语言基础上发展而来，但它比 C 语言更易于学习和掌握。C++以其独特的语言机制在计算机科学的各个领域中得到了广泛的应用。面向对象的设计思想是在原来结构化程序设计方法基础上的一个质的飞跃，它完美地体现了面向对象的各种特性。

(6) Java 语言最早诞生于 1991 年，起初被称为 OAK 语言，是 SUN 公司为一些消费性电子产品而设计的一个通用环境，是一种简单、跨平台、面向对象、分布式、健壮安全、结构中立、解释型、可移植、动态、性能很优异的多线程语言。

(7) C＃是一种安全、稳定、简单的，由 C 和 C++衍生出来的面向对象编程语言。它在继承 C 和 C++强大功能的同时去掉了它们的一些复杂特性，综合了 VB 简单的可视化操作和 C++的高运行效率，以其强大的操作能力、优雅的语法风格、创新的语言特性和便捷的面向组件编程的支持，成为. NET 开发的首选语言。

1.1.3 语言处理程序

计算机不能直接执行用高级语言编制的程序(源程序)，那么，计算机怎样鉴别源程序，确定它的正确性？如何运行并获得预期的计算结果呢？这些是语言处理程序要解决的问题。

用高级语言(或汇编语言)编写的程序(即源程序)，是语言处理程序要处理的对象，语言处理程序把源程序翻译成语义等价的计算机能够识别的低级语言程序(目标程序)。由此可见，对语言处理程序来说，源程序是其处理对象(输入程序)，目标程序是其处理结果(输出程序)，它是在高级语言和计算机之间起到翻译作用的程序。

语言处理程序可用汇编语言或高级语言写成。它是为用户设计的编程服务软件，其作用是将高级语言源程序翻译成计算机能识别的目标程序。语言处理程序环境一般由汇编程序或编译程序或解释程序与相应地操作程序等组成，相应地一般称为编译(程序)系统或解释(程序)系统等。

在软件领域中，包含程序设计语言和相关语言处理程序及其他相关工具的程序设计环境，通常被称为某种程序设计语言的集成开发环境。不同语言的语言处理程序是不同的，同一种高级语言也可能存在不同级别、不同版本的语言处理程序。程序员可以在不了解语言处理程序的情况下，借助集成开发环境完成程序的编制和运行。常用集成开发环境有多种，如微软的 Visual Studio. NET 是 C++、VB、C＃等语言的集成开发环境，而 C++ Builder、Delphi、JBuilder 分别为 C++、Pascal、Java 语言的集成开发环境。

1.1.4 设计程序的基本原则

程序设计也是一种工程设计(通常称为软件工程)。程序设计追求的目标是可靠性、可

理解性、可维护性和执行效率，但执行效率与可维护性、可理解性一般是相互矛盾的。因此程序设计的目标是设计出可靠、易读而且代价合理的程序。

设计程序时应该遵循的以下几个基本原则。

(1) 正确性：正确可靠无疑是设计程序最基本的要求，程序设计错误将导致设计程序工作毫无意义。

(2) 有效性：一个好的程序运行效率要高，既要考虑减少计算机运行时间，又要考虑节省计算机存储空间。

(3) 鲁棒性：程序设计时必须考虑到可能发生的异常事件并做出相应处理。一个好的程序应当是鲁棒的、健壮可靠的。即使用户非法操作(如不正确的输入)，也能继续正常工作。

(4) 可理解性：程序设计应选择恰当组织方式及布置格式，正确地编写、组织、管理与程序有关的程序清单、说明书、使用手册等文字资料，来提高它的可理解性。

(5) 可维护性：是指纠正程序出现的错误和缺陷，以及为满足新的要求进行修改、扩充或压缩的容易程度。

(6) 可移植性：简单地说就是指源代码在不同的平台上工作的兼容性。当程序移植到不同平台上，修改的代码越少，程序可移植性越好。

1.2 算　　法

1.2.1 算法的概念

算法(Algorithm)是为解决某一个具体问题而采取的确定的、有限的方法和步骤。计算机算法即计算机能够执行的算法，是指以一步接一步的方式来详细描述计算机如何将输入转化为所要求的输出的过程。通常计算机算法分为以下两类。

(1) 数值计算算法：是用数学的计算方法来得到运算结果。

(2) 非数值计算算法：则常用逻辑运算或数据运算来分析解决一些事务管理的问题。

在学习数学过程中，要解一个方程需要用一定的方法和步骤(算法)。下面通过求解一个数学题目来理解算法的概念。

例 1-1　求一个一元二次方程的解。

使用自然语言表示算法如下。

步骤 1：将方程设为标准形式，即 $ax^2+bx+c=0$。利用 b^2-4ac 的值来判断方程解的情况，即无解、有一个解或有两个解。

步骤 2：求 b^2-4ac 的值进行判断，如果该值小于 0，则此方程无解，执行步骤 5；如果等于 0，则此方程有一个解，执行步骤 3；否则方程有两个解，执行步骤 4。

步骤 3：求 $x=\dfrac{-b}{2a}$ 的值，为方程唯一的解，执行步骤 5。

步骤 4：求 $x_1=\dfrac{-b+\sqrt{b^2-4ac}}{2a}$，$x_2=\dfrac{-b-\sqrt{b^2-4ac}}{2a}$ 的值，为方程的两个有效解，执行步骤 5。

步骤 5：解题结束。

以上是在求解一元二次方程时所采用的方法和步骤。程序设计也是要利用类似方法和步骤控制计算机来解题或完成某项工作。这种方法和步骤就是计算机程序设计的算法。人们根据经验将一些典型和成熟的算法汇编成册，供学习和使用，但是算法设计的步骤和方法并不是固定不变的。如例 1-1 的求解算法中，也可以把步骤 2、3、4 顺序和内容做一些调整，照样可正确求解，这说明完成某一任务，并不一定只有唯一一种算法。但算法应具备一些特征。

一个算法应该具备以下五个特征。

(1) 有穷性(有限性)。任何一种算法都应在有限的操作步骤内可以完成，哪怕提出的解题方法是失败的。

(2) 确定性(唯一性)。解题算法中的任何一个操作步骤都应是清晰无误的，不会产生歧义或者使人误解。

(3) 可行性(能行性)。解题算法中的任何一个操作步骤在现有计算机软硬件条件下和逻辑思维中都能够实现。

(4) 有 0 到多个输入。解题算法中可以没有数据输入，也可以同时输入多个需要算法处理的数据。

(5) 一个算法执行结束之后必须有数据处理结果输出，哪怕是输出错误的数据结果，没有输出的算法是毫无意义的。

1.2.2 算法的表示方法

虽然算法与计算机程序密切相关，但二者也存在区别：计算机程序是算法的一个实例，是将算法通过某种计算机语言表达出来的具体形式；同一个算法可以用任何一种计算机语言来表达。描述算法可以用自然语言、程序流程图、PDL 图以及 N-S 图、伪代码、计算机语言等。下面介绍描述算法的几种常用方法。

1. 用自然语言描述算法

用日常使用的语言来描述算法，称为算法的自然语言描述。如例 1-1 中就是采用自然语言描述算法的方式。为加深对自然语言描述算法的理解，下面再看一个例子。

例 1-2 用自然语言描述求解 $S=1\times2\times3\cdots\times(n-1)\times n$ 的算法。

(1) 确定 n 的一个值；

(2) 假设 i 的初始值为 1；

(3) 假设 S 的初始值为 1；

(4) 如果 $i\leqslant n$ 时，转去执行⑤，否则转去执行⑦；

(5) 计算 S 乘以 i 的值后，重新赋值给 S；

(6) 计算 i 加 1 的值，然后将该值重新赋给 i，转去执行④；

(7) 输出 S 的值，算法结束。

从上面这个描述求解的过程中不难发现，自然语言描述算法的方法比较容易掌握。但是，存在很大缺陷，如果算法中含有多分支或循环操作，则很难表述清楚。此外，使用自然语言描述算法还很容易造成歧义。

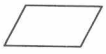
2. 用程序流程图描述算法

程序流程图(Program Flow Chart)是软件开发者最熟悉的一种算法表达工具，它独立

于任何程序设计语言。它的优点是直观、清晰、易于掌握，便于转化成任何计算机程序设计语言。因此，它是软件开发者常用的算法表示方式。

在学习使用流程图描述算法之前，先对流程图中的一些常用符号作一解释，如表 1-1 所示。

表 1-1　程序流程图的常用符号

符　号	名　称	作　用
	开始框或结束框	表示算法的开始或结束
	输入框或输出框	表示算法过程中，从外部获取的信息(输入)，然后将处理过的信息输出
	处理框	表示算法过程中，需要处理的内容，只有一个入口和一个出口
	判断框	表示算法过程中分支结构的条件，通常用菱形框中上面的顶点表示入口，其余顶点表示出口
	流程线	算法过程中流程指向的方向

例 1-3　用程序流程图描述例 1-2 求解过程的算法。

程序流程图如图 1-1 所示。

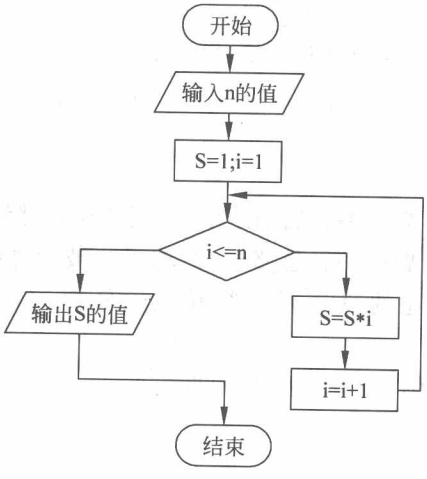


图 1-1　用程序流程图描述例 1-2 的算法

从上面算法流程图中，可以比较清晰地看出求解问题的执行过程。但是程序流程图的符号在使用过程中不容易规范，特别是在标准中没有严格规定流程线的用法，流程线能够指示流程控制方向的随意转移，很容易造成算法中操作步骤的执行次序混乱，而且不便于开发人员交流。

3. 用 N-S 图描述算法

N-S 图是 1973 年由美国学者 I. Nassi 和 B. Shneiderman 提出的一种新的流程图形式，N 和 S 是两位学者姓氏的首字母。在这种流程图中，摒弃了带箭头的流程线。算法的具体