“十二五”普通高等教育本科国家级规划教材

# C语言程序设计

（第3版）

黄维通 解辉 李祁 编著

高等教育出版社

五”普通高等教育本科国家级规划教材

# C语言程序设计

（第3版）

黄维通 解辉 李祁 编著

高等教育出版社·北京

内容提要

本教材是“十二五”普通高等教育本科国家级规划教材，从C语言程序设计的基本思想出发，以“基本概念-基本应用-能力培养”为主线，注重案例驱动与算法的应用与实现，强调程序设计应用开发能力的培养。

本教材主要内容包括程序设计中的基本概念与应用，如变量、数组、控制结构及条件判断结构等，适时引入函数的结构与应用、指针的概念及其应用、算法设计与实现、结构型数据的应用及文件的操作等面向应用的知识点，最后介绍了数据结构中链表的概念与应用。

本教材可作为高等院校、计算机水平考试、各类成人教育等教学用书，也可作为计算机爱好者的自学参考书。

**图书在版编目（CIP）数据**

C语言程序设计/黄维通，解辉，李祁编著. --3版. --北京：高等教育出版社，2018.12

ISBN 978-7-04-051058-4

Ⅰ. ①C… Ⅱ. ①黄… ②解… ③李… Ⅲ. ①C语言-程序设计-高等学校-教材 Ⅳ. ①TP312.8

中国版本图书馆CIP数据核字（2018）第259079号

策划编辑 刘 茜　　责任编辑 刘 茜　　封面设计 李卫青　　版式设计 童 丹

插图绘制 于 博　　责任校对 高 歌　　责任印制 赵义民

出版发行 高等教育出版社
社　　址 北京市西城区德外大街4号
邮政编码 100120
印　　刷 中国农业出版社印刷厂
开　　本 850mm×1168mm 1/16
印　　张 19.25
字　　数 440千字
购书热线 010-58581118
咨询电话 400-810-0598

网　　址 http://www.hep.edu.cn
http://www.hep.com.cn
网上订购 http://www.hepmall.com.cn
http://www.hepmall.com
http://www.hepmall.cn
版　　次 2003年5月第1版
2018年12月第3版
印　　次 2018年12月第1次印刷
定　　价 45.00元

本书如有缺页、倒页、脱页等质量问题，请到所购图书销售部门联系调换

物 料 号 51058-00

# C语言程序设计

第3版

黄维通 解辉 李祁

1 计算机访问http://abook.hep.com.cn/1852135，或手机扫描二维码、下载并安装Abook应用。

2 注册并登录，进入“我的课程”。

3 输入封底数字课程账号（20位密码，刮开涂层可见），或通过Abook应用扫描封底数字课程账号二维码，完成课程绑定。

4 单击“进入课程”按钮，开始本数字课程的学习。


课程绑定后一年为数字课程使用有效期。受硬件限制，部分内容无法在手机端显示，请按提示通过计算机访问学习。

如有使用问题，请发邮件至abook@hep.com.cn。


电子教案

习题解析

视频讲解

# ○ 前　　言

为适应新形势下计算机基础教育的改革，推进计算机程序设计基础课程以及配套的精品教材建设，从能力培养的理念出发，结合信息化社会对高素质、复合型人才的程序设计能力的需求，特出版此教材，本教材为《C 语言程序设计（第 2 版）》的修订版。本教材配套了电子教案、习题解析与视频讲解等内容，帮助读者进一步提高学习效果。

C 语言是目前国内外广泛使用的程序设计语言之一，也是广泛开设的计算机程序设计基础入门课程之一。C 语言具有功能丰富、使用方便灵活、程序执行效率高、可移植性好等特点，因此得到业内人士的青睐。它既具有高级语言的特点，又具有汇编语言的特点，系统处理能力较强。它支持自顶向下逐步求精的程序设计技术，其函数式结构为实现程序的模块化设计提供了强有力的保障。因此，它被广泛应用于系统软件和应用软件的开发。

本教材具有如下三个特点。

（1）通俗易懂，突出“三基”（基本概念、基本原理与基本应用）的介绍与应用。

（2）在介绍“算法设计与实现”时，重点介绍经典的排序与查找的算法及其实现，同时通过几种不同算法的比较，讨论算法的效率及代码实现的效率。通过介绍算法设计及其实现，可以使读者更好地学习程序设计的思想、体系结构和方法，尤其是优化的程序设计方法。

（3）为了用最简洁的语言讲解代码结构及功能，本教材中的例题代码中给出了详细的代码注释，有利于读者更好地理解代码。同时，教材中的所有代码均在 Visual Studio 2012 环境下调试通过。

本教材由黄维通、解辉和李祁编写，其中解辉编写第 1、2、3 章内容，黄维通和李祁共同编写第 4-10 章内容。本教材主要用于大学的程序设计基础课程的教学。为了更好地建设该教材，我们从读者的学习体验出发，贴合学生的认知规律，专门请学完该课程的清华大学生命学院临床医学专业 65 班学生强佳祺对该教材从学生认知角度进行审读并提出修改建议。同时解辉和李祁对此教材进行了交叉审读，并提出了很好的修改意见，解辉验证了本教材的所有例题代码，刘智灵参加了例题视频的制作。在此对他们的辛勤工作表示感谢，同时感谢出版社对本书的出版所给予的大力支持。

由于作者水平有限，错误在所难免，恳请读者批评指正并通过作者电子信箱告知。作者邮箱：huangwt@ tsinghua. edu. cn。

黄维通

2018 年 6 月

# ○ 目　　录

## 第 1 章　C 语言的基本概念



## 第 2 章　C 语言程序的基本数据类型及其运算



## 第 3 章　C 语言程序基本控制结构及其应用

## 第 4 章　数组及其应用



## 第 5 章　函数及其应用



## 第 6 章　指　　针



## 第 7 章　排序与查找算法及其实现

## 第 8 章 结构体、联合体和枚举



## 第 9 章 文 件 操 作

## 第 10 章 链表及其应用

# 第 1 章
# C 语言的基本概念

# 1.1　程序设计语言的发展历史

自 1946 年第一台计算机问世以来，计算机科学的发展逐渐引起人们的重视，计算机科学的应用也越来越广泛。目前，随着计算机网络的普及，计算机的应用已经渗入了人们的日常生活中。

计算机系统是由软件系统和硬件系统两部分构成。对计算机硬件系统来说，若没有软件系统的支持，那么计算机硬件只能是一堆“废铁”，俗称“裸机”。那么现在各种信息是如何被计算机硬件识别并执行的呢？通俗意义上来说，计算机硬件就像是电灯开关，它只有“打开”和“关闭”这两种状态，因此对于计算机硬件来说，需通过 0 和 1 两种状态来识别。

### 1. 机器语言

前述提到的计算机硬件只有 0 和 1 两种状态，所以计算机硬件只能识别由 0、1 构成的信息。

机器语言是由 0 和 1 组成的二进制数序列，其特点是它只能直接被计算机硬件识别，所以它的执行速度快，执行效率高。但是由于所有的代码都由 0、1 组成，因此读写机器语言程序不直观，不便于理解。

### 2. 汇编语言

鉴于机器语言程序的读写和纠错过程太困难，人们便开发了带有简洁自然语言和符号的汇编语言。例如用 ADD 表示加法，SUB 表示减法。但是计算机硬件是不能够直接识别这些符号，所以需要一个专门的翻译程序将这些符号翻译成为计算机硬件能直接识别的机器语言。

汇编语言的产生大大提高了程序编写效率。但是随着程序规模的逐渐增加，兼容性问题也随之出现，程序员发现在某机器上用汇编语言编写的程序，换到另一台机器的时候却无法正常执行。也就是说汇编语言依赖于具体的计算机硬件，移植性较差。但是由于汇编语言具有较高的执行效率，针对具体硬件编写的汇编程序能够较好地发挥作用，故在底层开发汇编语言仍是一种较好的开发模式。

### 3. 高级语言

鉴于汇编语言的不足，人们越来越期望出现一种程序设计语言，其既易学易懂，便于维护，又有较好的可移植性，在大部分计算机上可以通用。1954 年，人们期望中的这种语言终于问世了，那就是 Fortran 语言。在此后的几十年里，程序员们根据自己和软件的需求不断改进和创造新的语言，到目前为止，已有几百种语言问世，包括较为熟悉的 Basic、C、Visual Basic、Delphi、C#、C++、Java 等，这些语言被统称为高级语言。

一个程序设计思想贯穿于高级语言的发展，就是：面向过程→面向对象→面向组件。

- 面向过程

面向过程是传统的编程方式，程序按照预先设定好的顺序一步一步地执行。若要

编写一个程序来解决某个问题，前提是必须清楚解决这个问题的详细过程。计算机内部的任何操作都可以看作是对数据信息的处理，那么在面向过程的程序设计中就是对这些数据的操作。

- 面向对象

面向对象的程序设计思想是将程序设计过程更贴近于人类的思维方式，将现实世界中的某个事物看作是一个整体对象，面向对象的程序就是对这些对象进行处理。

在面向过程编程中，数据和操作是分开的，不同的程序使用不同的数据。在面向对象编程中，编程者需要定义对象和操作，所以对象数据和操作是一体的，这是面向对象和面向过程编程的重要差别之一。

- 面向组件

在中小规模的软件设计中，对象之间的相互协作关系就能满足需要。但是随着软件规模的扩大和其复杂性的增加，对象的数量也急剧增加，对象之间的协作关系也就越来越复杂。于是，程序员需要一种新的程序设计思想来解决这一问题，即面向组件编程思想。面向组件编程是对面向对象编程的补充，帮助软件实现更加优化的程序结构。

## 1.2 程序设计过程中的几个基本概念

### 1. 程序与程序设计

计算机程序是一组计算机能够理解并执行的指令序列，而程序设计则是给出解决特定问题程序的过程。

程序运行于某种目标体系结构之上，它控制着计算机的工作流程，完成一定的逻辑功能最终达到解决问题的目的。通常程序主要包括两方面的信息：对数据的描述和对操作的描述。前者是数据结构，即指在程序中要指定用到哪些数据以及这些数据的类型和数据的组织形式，后者则是算法，即计算机执行的操作步骤。因此，对程序的描述有如下等式：

$$\text{程序} = \text{数据结构} + \text{算法}$$

有人认为这个公式改写为以下形式更为恰当：

$$\text{程序} = \text{算法} + \text{数据结构} + \text{程序设计方法} + \text{编程语言}$$

其中，算法是处理问题的策略，数据结构是问题的数学模型，程序设计方法是程序高效性、合理性的保证，编程语言是基本工具。

### 2. 算法

算法是程序设计中一个非常重要的概念，所谓算法就是处理问题的方法和步骤。从计算机的角度讲，算法和程序都是指令的有限序列，正确的程序能够体现算法。算法与程序的区别有如下几点。

- 程序必须是被计算机理解且可执行的，而算法无此限制。
- 算法中描述的步骤一定是有限的，而程序可以无限地执行。

• 算法代表了对问题的解，而程序则是算法在计算机中特定地实现。一个算法若用程序设计语言来描述，则它就是一个程序。

一个完整的算法应具有如下特征。

• 有穷性：指解决问题的方法和步骤必须是有穷尽的，能在有限的执行步骤后给出正确结果并结束。例如，$N!=1\times2\times3\times\cdots\times(N-1)\times N$ （$N$ 是正整数）其中 $N$ 是一个特定数，那么这种描述就可以称为一个算法。而表达式：sum=1+2+3+…+$N$+…就不能称之为算法，因为该描述在执行有限步后仍不能结束，不满足“有穷性”的特点，它只能称为一个计算方法。

• 确定性：正确的算法要求组成算法的规则和步骤是唯一确定的，不能存在二义性和模糊性，二义性和模糊性都是不可操作的。而且算法指定的操作是有序的，必须按指定的顺序执行，并能够在有限的执行步骤后给出正确的结果。如果算法存在二义性，程序的编码工作将无法进行。同时，算法也不允许存在模糊的概念。如“加一吨水”有明确的概念，这在程序的计算或控制中是可以操作的，如果说“加一些水”，那就无法操作了。日常生活中，“一些”没有明确的意义，程序是根据人的具体要求来完成相应工作的。

• 有效性：针对一个任务给出的解决方法和步骤必须是有效的，在经过有限步骤的执行后能够给出正确的结果，否则算法是无意义的。

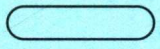
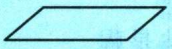
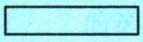
• 可执行性：可执行性是指算法的所有操作都是能通过计算机程序代码实现，又称可操作性。

• 零个或多个输入，至少一个输出：尽管大多数算法输入参数是必要的，但对于个别情况，如打印“hello world!”，不需要任何输入参数，因此算法的输入可以是零个。至少一个输出反映对数据加工后的结果，没有输出的算法是毫无意义的。

算法的常用描述方法有自然语言、程序流程图和伪代码，这里只介绍程序流程图。

程序流程图是利用一些程序框和流程线来描述算法，它可以比较清晰、直观地描述程序的控制流程，它不依赖于任何具体的计算机和程序设计语言，因此适用于不同环境的程序设计。如表 1-1 所示是流程图中主要元素的常用符号标识。

表 1-1　流程图的主要元素

| 程序框 | 名称 | 功能 |
| --- | --- | --- |
|  | 开始/结束 | 算法的开始和结束 |
|  | 输入/输出 | 输入和输出信息 |
|  | 处理 | 计算与赋值 |
|  | 判断 | 条件判断 |
|  | 流程线 | 算法中的流向 |

下面举例说明求 50! 的算法流程图，如图 1-1 所示。

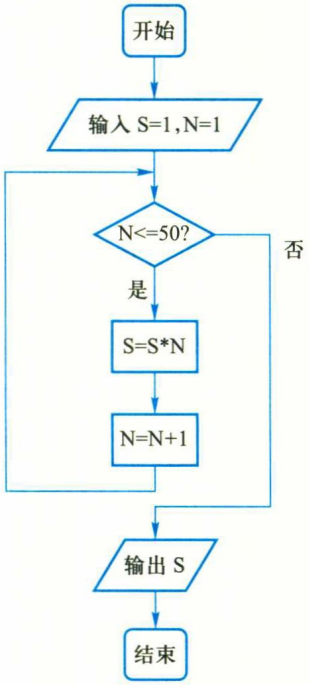



图 1-1 计算 50! 的流程图

### 3. 数据结构

数据结构是指相互之间存在一种或多种特定关系的数据元素的集合，包含 3 个要素：逻辑结构、存储结构和基本运算。算法和数据之间存在本质联系，一个良好的数据结构可以提高算法执行效率，反之算法又联系着数据在计算过程中的组织方式，所以算法也是研究数据结构的重要途径。

## 1.3 软件工程的概念

当今计算机技术的迅猛发展已使软件开发步入工程化阶段。一个大型软件的开发一般需要经历规划、需求分析、设计、编码、测试和运行维护等几个阶段。其中，正确的需求分析对程序设计是至关重要的，而编码就是对需求的具体实现。在编码过程中，算法十分重要，良好的算法往往会取得事半功倍的效果，算法思想决定了程序的质量和性能。

软件工程是开发、运行、维护和修复软件的系统方法。研究学者认为，软件工程就是运用现代科学技术知识来设计并构造计算机程序，即开发、运行和维护这些程序所必需的相关文件资料。软件工程包括三个要素：方法、工具和过程。

软件工程方法为软件开发提供了“如何做”的技术。它包括了多方面的任务，如项目计划与估算、软件系统需求分析、数据结构、系统总体结构设计、算法设计、编码、测试以及维护等。

软件工具为软件工程方法提供了自动或半自动的软件支撑环境。目前，已经开发出许多软件工具，足以支持上述的软件工程方法。而且已经有人把诸多软件工具集成起来，使得一种工具产生的信息可以为其他的工具所使用，这样建立起一种称为计算机辅助软件工程（computer aided software engineering，CASE）的软件开发支撑系统。CASE 将各种软件工具、开发机器和一个存放开发过程信息的工程数据库组合起来形成一个软件工程环境。

软件工程的过程则是将软件工程的方法和工具综合起来以达到合理、及时地进行计算机软件开发的目的。过程定义了方法使用的顺序、要求交付的文档资料、为保证质量和协调变化所需要的管理及软件开发各个阶段完成的内容。

软件工程就是包含上述方法、工具及过程在内的一些步骤。

开发一个大型应用程序，就必须从软件工程的思想出发，对需求进行系统性的分析与设计，而不是匆匆编码。今天读者开始学习基于 C 语言的程序设计，只是培养编程能力的第一步，要把自己培养成合格的程序员或系统分析员，就必须牢固树立软件工程的思想，系统、全面地看待一个细小的问题，这样才能更好地掌握编程技术和方法，有关软件工程的更多知识，请读者阅读相应的资料。

## 1.4　C 语言的发展与特点

### 1.4.1　C 语言的发展和 ANSI C 标准

C 语言是目前国际上广泛流行的一种基础的结构化程序设计语言，它不仅是很好的开发系统软件的工具，也是开发应用软件的程序设计语言。因此，它深受广大程序设计者的欢迎。

C 语言是在 20 世纪 70 年代初由美国贝尔实验室的丹尼斯·里奇（Dennis Ritchie）设计的，并首先在安装 UNIX 操作系统的 PDP-11 计算机上实现，因此，最初的 C 语言是为了描述和实现 UNIX 操作系统而设计的。1973 年，C 语言的主体功能完成，Thompson 和 Dennis 两个人合作把 UNIX 的 90%以上内容用 C 语言进行了改写，即人们熟知的 UNIX 第 5 版。多年来，UNIX V 系统配备的 C 语言一直是公认标准，在 Brian Kernighan 和 Dennis Ritchie 合著的 *C Programming Language* 中对此也进行了介绍。随着微型计算机的普及，出现了较多的 C 语言系统，其中多数系统接受的源程序都能高度兼容，然而，没有统一的标准，各系统之间总存在一定的差异，这种差异对于计算机应用技术的发展显然是不利的。

所谓标准就是将已存在的实践内容整理成文，让大家使用。编程标准就是软件开发人员和编译器设计人员之间的协议。这份协议包括两部分，一部分是开发人员需要且编译器设计人员同意提供的内容，另一部分就是开发人员愿意遵守且编译器设计人员认为应该遵守的规则。

为了克服多个 C 语言系统没有统一标准的不利局面，ANSI（美国国家标准研究

所）于 1983 年成立了专门成立了 C 语言标准委员会，花了 6 年时间使 C 语言迈向了标准化。随着 C 语言的广泛应用又不断推出新的 C 语言版本，其性能也越来越强。1975 年，随着 UNIX 第 6 版的推出和面向对象程序设计技术的出现，C 语言的突出优点引起了人们的普遍关注，1989 年，ANSI C 语言标准被采用，该标准被称为 ANSI C 标准或 C89。该标准中详细说明了使用 C 语言书写程序的形式，规范了对程序的解释，包括以下几点。

（1）C 程序的表示法；

（2）C 语言的语法和约束；

（3）解释 C 程序的语义规则；

（4）C 程序输入和输出的表示；

（5）一份标准实现的限定和约束。

到了 1995 年，出现了 C 语言的修订版，其中增加了一些库函数，出现了初步的 C++，在此基础上，C89 成为 C++的子集。此后，C 语言不断发展，在 1999 年又推出了 C99，C99 在基本保留了 C 的特性的基础上增加了一系列新的特性。随后又几经修改和完善，它也从面向过程的编程语言发展到面向对象的程序设计语言。

### 1.4.2 C 语言的特点

C 语言之所以能被广泛使用，主要因为 C 语言的如下几个特点。

#### 1. 易学易用

C 语言简洁、紧凑，程序书写形式符合人类认知规律，语法逻辑性较强，较为自由，适合初学程序设计的人员学习使用。而且编程中的关键字基本上都能跟相应含义的英文单词或缩写联系起来，而那些单词一般很简单。

#### 2. 结构化语言

C 语言是便于进行模块化程序设计的语言。C 语言程序是由一系列函数组成，这种结构便于把一个大型程序划分为若干个功能相对独立的模块，模块间通过函数调用来实现相互连接。可以通过函数使得一个程序中的各个任务被分别定义和编码，从而使程序模块化。一个设计良好的函数可以在各种情况下正常工作，不会对程序的其他部分产生不良影响。因此，对于大型程序的设计来说，函数的设计是至关重要的，因为函数是 C 语言的独立的子程序或功能模块，它是一种构件，程序的所有操作都在其中发生。

C 语言能够把执行某个特殊任务所需要的指令和数据从程序的其余部分分离并隐藏，实现代码和数据地封装。结构化语言还提供了大量的程序设计功能，直接支持顺序、分支和循环三种典型的基本结构（这三种结构将在第 3 章进行详细介绍），使程序设计人员便于使用“自顶向下逐步求精”的结构化程序设计技术。

#### 3. 可移植性

C 语言程序具有较高的可移植性。可移植性指的是，可以把为某种计算机编写的软件经过很少的修改或不修改就能运行在另一种机器或操作系统上。C 语言不包含依赖硬件的输入输出机制，其输入输出功能是由独立于 C 语言的库函数来实现的。这样就使 C 语言程序本身不依赖于硬件系统，也便于在不同的机器系统间移植。

# 1.5　C 语言程序的基本字符集和标识符

任何一种高级语言，都有自己的基本词汇符号和语法规则，程序代码都是由这些基本词汇符号（有时也称为标识符）并根据该语言的语法规则编写而成，C 语言也不例外。C 语言规定了其所需的基本字符集和标识符。

## 1. 字符集

满足 C 语言文法要求的字符集如下：

（1）英文字母 a~z，A~Z。

（2）阿拉伯数字 0~9。

（3）特殊符号如下所示。

| | | | | | |
|---|---|---|---|---|---|
| + | - | * | / | % | = |
| { | } | ( | ) | [ | ] |
| _（下划线） | '（单引号） | . | : | ? | ~ |
| < | > | & | ; | " | \| |
| ! | # | 空格 | ^ | | |

## 2. 标识符

C 语言的标识符主要用来表示常量、变量、函数和类型等的名字，是只起标识作用的一类符号，标识符由下划线或英文字母构成，它包括如下 3 类。

（1）保留字

每一个保留字都有特定含义，不允许用户把它们当作变量名（关于变量及变量名的概念，将在第 2 章进行介绍）使用，C 语言的保留字都用小写英文字母表示，常用的保留字如下所示。这些保留字的含义及用法，将在后续章节中随着编程技术介绍的深入而逐步接触到。

| | | | | | |
|---|---|---|---|---|---|
| auto | break | case | char | const | continue |
| default | do | double | else | enum | extern |
| float | for | goto | if | int | long |
| return | short | signed | static | struct | switch |
| typedef | union | unsigned | void | volatile | while |

（2）预定义标识符

除了上述保留字外，还有一类具有特殊含义的标识符，它们被用作库函数名和预编译命令，这类标识符在 C 语言中称为预定义标识符。一般来说不要把标识符再定义