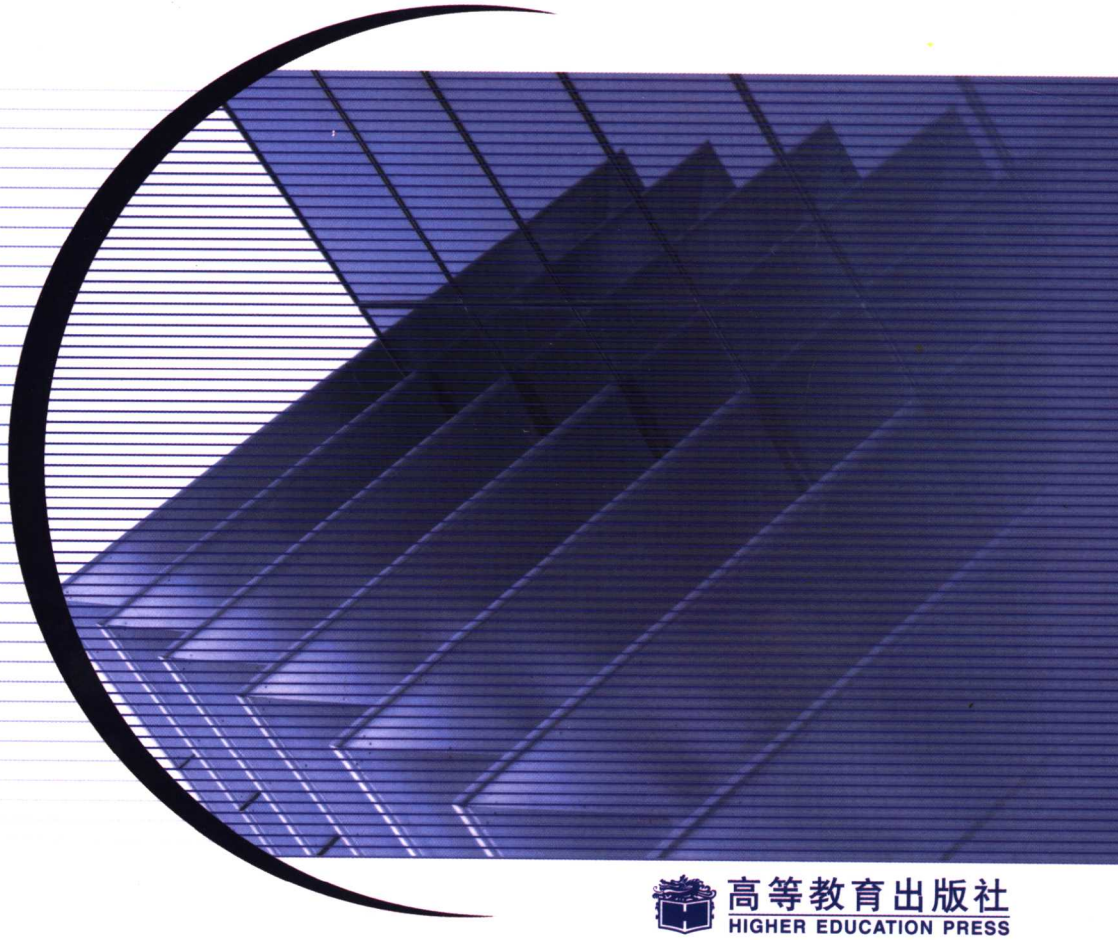
高等学校计算机基础教材

C语言程序设计教程

第3版

谭浩强　张基温　编著



高等教育出版社
HIGHER EDUCATION PRESS

内容提要

在本书第一版（1992年出版）和第二版（1998年出版）已经发行超过100万册之后，根据国家“十一五”教材规划，著名计算机教育家谭浩强教授和张基温教授又通力合作，全面修订，推出了本书第3版。第3版保留了深入浅出、易学易懂、取舍合理、适宜教学的特点，并进一步充实了内容，在叙述方面更加详尽和深入。本书是以C语言的最新标准C99为依据编写的。

学习程序设计必须重视实践环节，多读程序，多编写程序，多上机实践。本书提供了大量典型的例题分析和用于自测的丰富习题；配有《C语言习题集与上机指导（第3版）》和教学课件，为教与学创造了一个立体的环境。

本书可以作为高等学校计算机及相关专业学生学习C语言程序设计的教材，也可作为参加有关考试和自学的参考书。

本书配套教学资源可以在高等教育出版社高等理工教学资源网下载，网址为http://www.hep-st.com.cn。

图书在版编目（CIP）数据

C语言程序设计教程/谭浩强，张基温编著. —3版.
北京：高等教育出版社，2006.8
ISBN 7-04-019910-6

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

中国版本图书馆CIP数据核字（2006）第079966号

策划编辑 雷顺加　**责任编辑** 康兆华　**封面设计** 王凌波　**责任绘图** 尹　莉
版式设计 范晓红　**责任校对** 姜国萍　**责任印制** 毛斯璐

出版发行	高等教育出版社	**购书热线**	010-58581118
社　　址	北京市西城区德外大街4号	**免费咨询**	800-810-0598
邮政编码	100011	**网　　址**	http:// www.hep.edu.cn
总　　机	010-58581000		http:// www.hep.com.cn
		网上订购	http:// www.landraco.com
			http:// www.landraco.com.cn
经　　销	蓝色畅想图书发行有限公司	**畅想教育**	http:// www.widedu.com
印　　刷	唐山市润丰印务有限公司		
		版　　次	1991年8月第1版
			2006年8月第3版
开　　本	787×1092　1/16	**印　　次**	2006年8月第1次印刷
印　　张	21	**定　　价**	25.00元
字　　数	510 000		

物料号　19910-00

郑重声明

前　言

早在20世纪90年代初，C语言就表现出强劲的生命力。当时，为了把这种充满希望的程序设计语言引入中国高等学校的课堂，我们编写了这本教材。迄今它的发行总量已经超过100万册，成为备受广大读者青睐的一本教材。十几年过去了，在这期间，C语言的标准已经推进到C99；我们自己对C语言及其程序设计的理解也在不断升华；随着C语言程序设计教学的广泛展开，使用本书的教师和广大读者也有了新的要求，并提出了许多很好的建议。在这三种动力的推动下，我们决定对它进行一次比较大的修订。

程序设计是一种技术，也是一项工程。作为一本程序设计教材，不仅要介绍关于C语言的基本语法知识，还要强调思维方法的培养并着眼于应用现代软件工程思想进行程序开发能力的训练。如何解决好这3个方面的衔接，将它们有机地结合起来，是当前程序设计教材需要解决的一个重要问题，也是一个难点问题。为此，我们花费了20多年的时间，进行了各种探索。我们的一些经验和体会也融入了本书第3版的修订中。

基于算法思维方法的训练是现代信息素养的重要内容。程序设计课程的教学是引导学习者利用计算机进行解题的能力培养过程。本书选择了比较典型的问题，强调对问题的分析过程，目的在于通过典型问题分析，使读者能够举一反三，不断积累解决复杂问题的能力。

程序设计语言是程序设计的工具。一种程序设计语言凝聚了具有时代特征的程序设计理念和方法。为了有效地进行程序设计，正确地应用程序设计语言表达算法，必须准确地运用程序设计语言，掌握其语法知识。但是，程序设计教材不是语法手册，不可能包含全部语法内容。本书所选择的语法知识只是进入C语言天地的一些基本知识。因为任何一本教材都不可能是万能的，不可能既适合初学者，又适合需要进一步提高者。我们仅仅把这本教材定位在刚刚开始涉猎C语言程序设计的初学者。

教材是知识和能力的传播载体，它与其他科技著作不同，不仅要介绍有关知识，还要充分考虑到让读者喜欢和容易学习。因此，在本书修订时，我们一方面注意概念准确，科学性强，还特别注意了如何让学习者（包括一些自学者）感到容易学习，容易理解。

本书程序都是在Visual C++编程环境下进行调试的，在其他C语言环境下基本上都可以运行通过。

我们同时还对配套书《C语言习题集与上机指导》进行了修订，使课堂教学、实验和实践之间的关系更为密切。同时，还提供了相关多媒体教学课件素材，形成一套立体化的教学资源。

这次修订是在两位作者共同讨论的基础上，主要由张基温执笔，谭浩强修改、审定并统稿而成。参加本书编写工作的还有唐永炎、董兆军、段富、贾中宁、袁玫、谭亦峰、张秋菊、赵彦、钟实等。

在本书付梓出版之际，我们衷心感谢为本书提出过中肯建议和做过贡献的同仁和读者。

学海无涯，教海更无涯。本书出版以后，也还会存在许多不足。我们恳切地期望得到更多的批评和建议，以便把本书修改得更好。

谭浩强　张基温

2006年4月18日

目　录

第 1 章

C 语言程序设计的概念

1.1 程序与程序设计语言

1.1.1 计算机与程序

一提起计算机，大多数人会联想到主机、显示器和键盘。其实，应用要早得多、并一直流传至今的计算工具是算盘（如图 1.1 所示）。那么，现代电子计算机与算盘之间的最大区别在哪里呢？关键在于现代计算机可以自动完成计算过程，而算盘所进行的计算过程必须在人为拨动下才能实现。

那么，为什么现代计算机能够自动完成计算过程呢？这首先要从程序说起。程序实际上是一个非常普通的概念：按照一定顺序安排的工作步骤。可以说，完成任何事情都有相应的程序。所完成的事情不同，要求的效果不同，程序就不同。例如，针对同样的原料，采用不同的程序，会烹制出不同的菜肴。

一种工具若能自动工作，一是要有记忆功能，能够记住程序；二是具有按照程序控制相关部件操作的能力。如果能让算盘记住某种算法口诀和计算所需用到的数据，并且具备按照口诀控制算珠自动运动的机制，那么只要发出开始执行的命令，算盘就会自动完成计算。

遗憾的是，这样的机制并未在算盘中实现。但是，却有另外一种机器在这方面向前推进了一步，这就是明朝末年宋应星在其《天工开物》中记载的中国古代提花机（如图 1.2 所示）。

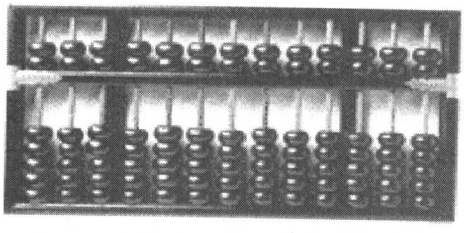


图 1.1　算盘

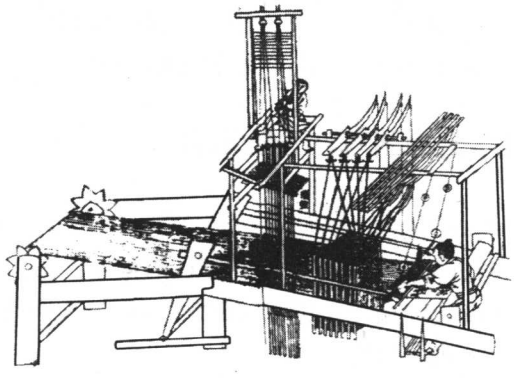


图 1.2　中国古代提花机

中国提花机大约出现于西汉末年（公元前）。它采用丝线所结成的“花本”（花版）控制经线起落，以织成所要求的图样。这是最早的程序控制思想。后来，提花机沿着丝绸之路传至欧洲，历经多次改进。1805 年，法国人 Joseph Jacquard 制造成功用穿孔卡片（如图 1.3 所示）控制连杆（横针）、用有孔/无孔进一步控制经线起落的提花机。

穿孔卡片把程序控制技术向前推进了一步。这一技术被英国数学家 Charles Babbage（如图 1.4 所示）引入计算机，通过有孔和无孔的组合来表示数据和程序。

18 世纪末，法国数学界召集大批数学家，组成了人工手算的流水线，经过长期艰苦奋斗，终于完成了 17 卷《数学用表》的编制。但是，手工计算出的数据出现了大量错误。这件事情强烈刺激了 Babbage。1812 年，20 岁的 Babbage 开始计算机的研制工作，他要把函数表的复杂算式转化为差分运算，用简单的加法代替平方运算，快速编制不同函数的数学用表，并将这种机器称为“差分机”。经过 10 年的努力，最终于 1822 年完成了第一台差分机，可以处理 3 个不同的 5 位数，计算精度达小数点后 6 位。1833 年，他又开始投身于一种“会分析的机器”——分析机的研制之中。他把机器设计成 3 个部分。一是用来存储数据信息的“仓库”（The Store），二是进行数据运算处理的“工场”（The Mill），三是使用穿孔卡片来输入程序并用穿孔卡片输出数据。

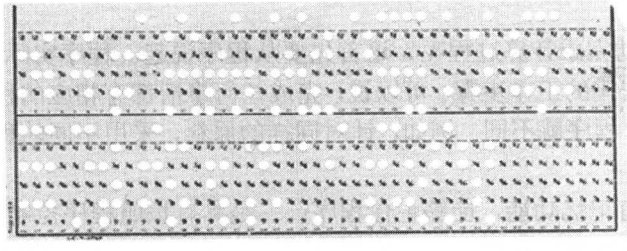


图 1.3　穿孔卡片

图 1.4　英国数学家 Charles Babbage

这台机器虽然没有研制成功，但其工作原理——程序存储控制为当今的计算机奠定了基础。

（1）任何工具所做的工作，都是由程序控制的。

（2）只有工具具备记忆程序的功能，并能按照程序进行自我控制，该工具才能自动工作。

1.1.2　计算机程序设计语言

程序需要用某种形式（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，珠算的语言是口诀。现代计算机的程序则是用计算机程序设计语言来描述的。从计算机诞生至今，程序设计语言也在伴随着计算机技术的进步不断发展。

1．机器语言

CPU 指令系统（也称该 CPU 的机器语言）是该 CPU 可以识别的一组由 0 和 1 序列所构成

的指令码。下面是某 CPU 指令系统中的两条指令：

10000000 （进行一次加法运算）

10010000 （进行一次减法运算）

用机器语言编写程序，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，却由于其不够直观、难记、难认、难理解、不易查错而只能被少数专业人员所掌握，同时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（使用穿孔纸带的时期）编程（用有孔、无孔分别代表 1、0），现在已经不再使用。

2. 汇编语言

为了减轻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一些“助记符号”来代替 0、1 码进行编程。如前面的两条机器指令可以写为

A+B ⇒ A 或 ADD A，B

A–B ⇒ A 或 SUB A，B

这种用助记符号来进行描述的指令系统称为符号语言或汇编语言。

用汇编语言编写程序，程序的生产效率和质量都有所提高。但是，汇编语言指令是机器所不能直接识别、理解和执行的。用它编写的程序经检查无误后，要先翻译成机器语言程序才能被机器所理解和执行。这个翻译转换过程称为“代真”。代真后所得机器语言程序称为目标程序（object program），代真之前的程序称为源程序（source program）。由于汇编语言指令与机器语言指令基本上存在一一对应关系，所以汇编语言源程序的代真可以由汇编器以查表方式进行。

汇编语言与机器语言均依 CPU 的不同而异，它们都称为面向机器的语言。用面向机器的语言编写程序，可以编出执行效率极高的程序。但这要求程序员不仅要考虑解题思路，还要熟悉机器的内部结构，并能“手工”地进行存储器分配。这种编写程序方法的劳动强度仍然很大，给计算机的普及推广造成很大障碍。

3. 高级语言

汇编语言和机器语言是面向机器的，不同类型计算机所使用的汇编语言和机器语言是不同的。1954 年出现的 FORTRAN 语言以及随后相继出现的其他高级语言，开始使用接近人类自然语言、但又消除了自然语言中的歧义性的语言来描述程序。这些高级语言使人们开始摆脱进行程序设计必须先熟悉机器内部结构的桎梏，把精力集中于解题思路和方法上。

第一种高级语言是 1954 年问世的 FORTRAN 语言。此后不久，不同风格、不同用途、不同规模、不同版本的面向过程程序设计语言便风起云涌。据统计，全世界已有 2 500 种以上的程序设计语言，其中使用较多的有近百种。如图 1.5 所示为几种广泛流行的高级语言的发展变迁情况。

1.1.3 高级语言程序的开发过程

一般来说，高级语言程序开发的一般过程包括如图 1.6 所示的几个步骤。

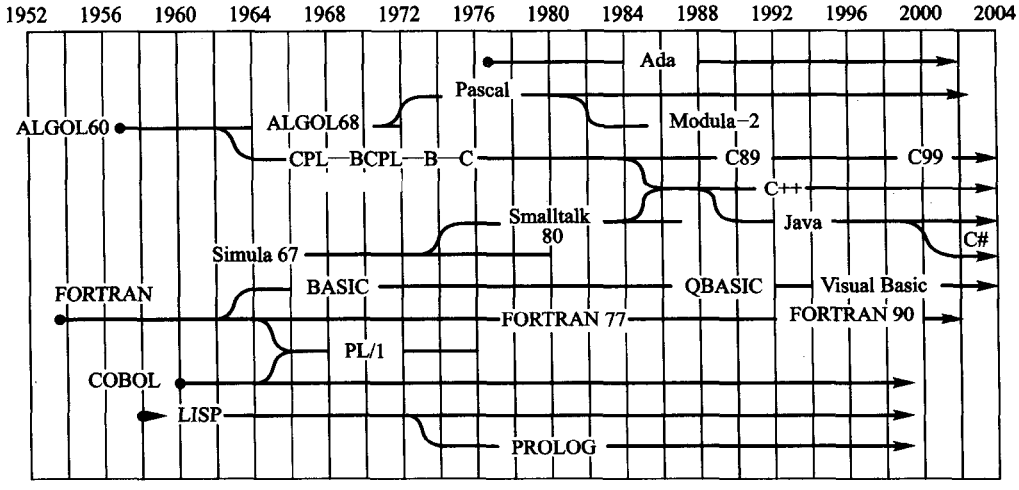


图 1.5 几种广泛流行的高级语言的发展变迁情况

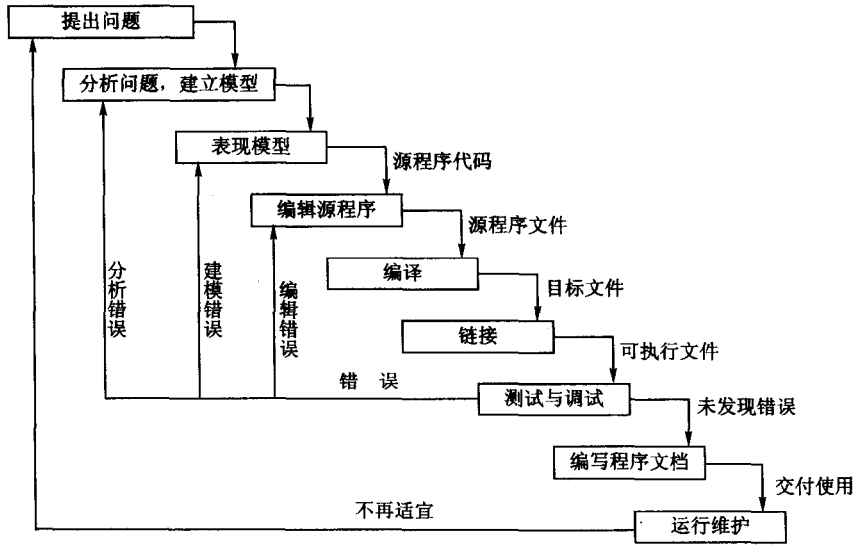


图 1.6 高级语言程序的开发过程

1．分析问题，建立模型

一般来说，一个具体的问题要涉及诸多方面，这是问题的复杂性所在。为了便于求解，往往需要忽略一些次要方面。这种通过忽略次要方面而找出解题规律的过程，就称为建立模型。

2．表现模型

表现模型是指用一种符号-语言系统来描述模型。一般来说，模型的表现会随着对问题抽象程度的加深和细化，不断由具有领域特色向计算机可解释、可执行的方向靠近，中间也可能采用其他一些符号系统，如流程图等，直到最后用一种程序设计语言描述出来。

3．源程序的编辑

源程序的编辑就是在某种字处理环境下，用具体的程序设计语言书写并对其进行修改的过

程。为此需要掌握一种程序设计语言。还要通过一种专用程序编辑器或通用的文字编辑器来实现。

4．程序的编译（或解释）与链接

编写出一个高级语言程序后，并非可以立即拿来执行。要让计算机直接执行程序，还要将它翻译成可由机器直接辨认并可执行的机器语言程序。为了对二者加以区分，把用高级语言编写的程序（文件）称为源程序（文件），把机器可直接辨认并可执行的程序（文件）称为可执行程序（文件）。这一过程一般分为以下两步。

（1）第 1 步

在程序编辑过程中，所输入源文件中的是一些字符码，但是机器能够直接处理的是 0、1 信息。为此，首先要将源程序文件翻译成 0、1 码所表示的信息，并用相应的文件保存。这种保存 0、1 码信息的文件称为目标程序文件。由源程序翻译成目标程序的过程称为编译。在编译过程中，还要对源程序中的语法和逻辑结构进行检查。编译任务是由称为编译器（compiler）的软件完成的。目标程序文件尚不能被执行，它们只是一些目标程序模块。

（2）第 2 步

将目标程序模块以及程序所需的系统固有目标程序模块（如执行输入/输出操作的模块）链接成一个完整的程序。经正确链接所生成的文件才是可执行文件。完成链接过程的软件称为链接器（linker）。

如图 1.7 所示为编译和链接过程示意图。

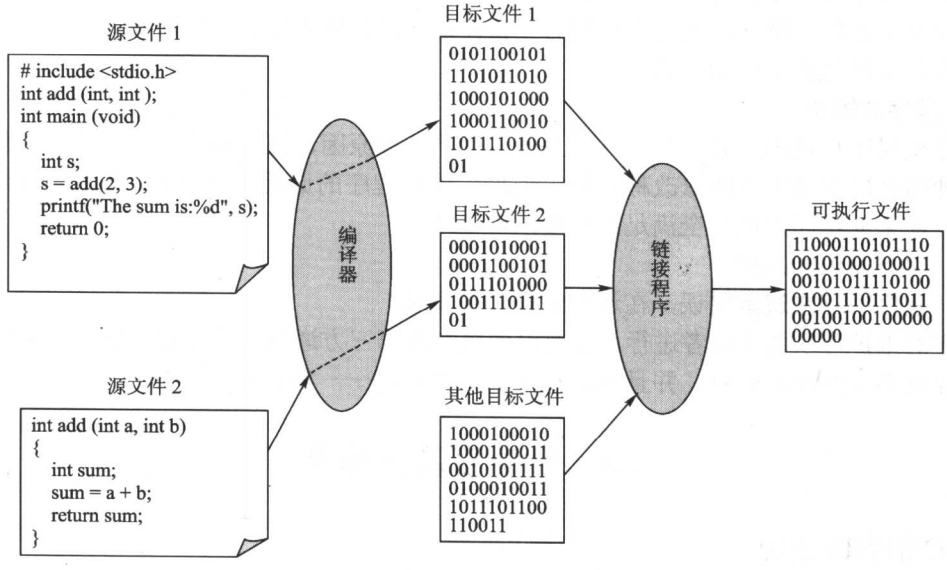


图 1.7 编译和链接过程示意图

程序在编译和链接过程中，也可能会发现错误。这时要重新进入编辑器进行编辑。

5．程序的测试与调试

程序文件经过编译和链接，生成可执行文件，就可以在计算机上执行了。但是，这并不意

味着可以得到预期的结果而交付用户使用，因为程序仍然会存在某些缺陷或错误。因此，每个人编写出一个程序后，在正式交付使用之前，总要试通一下。“试通”就是指试运行程序，也就是对程序进行测试。

测试是以程序通过编译、不存在语法和链接错误为前提，目的是找出程序中可能存在的错误并对其加以改正。因此，应该测试程序的运行情况，输入不同的数据可以检测出程序在不同情况下运行的状况。测试数据应是以“程序是会有错误的”为前提而精心设计出来的，并非随心所欲地拼凑而成。它不仅应含有被测程序的输入数据，还应包括程序执行后的预期结果。每次测试都要把实际结果与预期结果相比较，以观察程序是否出错。

6．编写程序文档

经过了问题分析、设计、程序编码、测试后，程序开发工作就基本结束了。但是，这时仍不能交付使用。因为随着程序规模的增大和日益复杂化，程序的使用和运行也越来越复杂，用户要运行程序，还需要知道许多其他信息。

（1）程序的功能

（2）所需输入数据的类型、格式和取值范围

（3）所需使用的文件数量、名称、内容以及存放位置等

（4）程序运行所需的软/硬件环境

（5）程序的装入、启动方法以及交互方式等

为此，程序开发者需要向用户提供一些资料——称为程序使用说明书或用户文档。需要说明的是，在许多软件中，这些内容已经部分或全部地以“readme”或“help”的形式提供。

目前，程序文档已经成为软件开发产品中的必需部分。文档在程序使用和维护过程中的重要性也改变了软件的概念，使之由早期的“软件是计算机程序的总称”演化为“软件是计算机程序以及计算机化的文档的总称”。

7．程序的维护

程序交付用户使用之后，并非万事大吉。由于多种原因，还可能需要对程序进行修改。程序交付使用之后对其进行的修改称为程序维护。维护程序的原因主要有以下几个方面。

（1）原来的程序未能完全满足用户要求。

（2）用户需求的改变。

（3）程序中存在遗留错误，在运行过程中被发现。

程序的维护可以由开发者进行，也可以由用户或第三方进行。为了便于进行程序维护，开发者应提供必要的技术资料，并且保证程序的可读性好——能让别人看懂。

1.2 C语言及其标准

1.2.1 C语言的出现

C语言是目前编程领域中最有影响力的一种程序设计语言。可是，它却是“漫不经心”地开发出来的。

在20世纪60年代，贝尔实验室的Ken Thompson（如图1.8所示）着手开发后来对计算机业界产生巨大影响的UNIX操作系统。为了描述UNIX，Thompson首先将当时的一种专门用来

描述系统程序的 BCPL 语言改进为他称之为 B 语言的语言。1970 年，Thompson 发表了用汇编语言和 B 语言所写成的 PDP-7 上实现的 UNIX 的初版。

1971 年，Dennis Ritchie（如图 1.8 所示）开始协助 Thompson 开发 UNIX。他对 B 语言做了进一步的充实和完善，加入新的数据类型和句法，于 1972 年推出一种新型程序设计语言——C 语言（取 BCPL 的第 2 个字母）。为了使 UNIX 操作系统得以推广，Ritchie 于 1974 年发表了不依赖于具体机器系统的 C 语言编译文本“可移植的 C 语言编译器”。由此可见，C 语言是借助 UNIX 操作系统的翅膀而起飞的，UNIX 操作系统也由于 C 语言而得以快速推广，二者相辅相成，成就了软件开发史上历时 30 年之久的一个时代。

图 1.8 Thompson（左）和 Ritchie（中）于 1999 年接受当时美国总统克林顿授予的国家技术勋章

1978 年，Brian W. Kernighian 和 Dennis M.Ritchie 合作出版名著 The C Programming Language，从而使 C 语言成为目前世界上流行最为广泛的高级程序设计语言。此后，又有多种程序设计语言在 C 语言的基础上产生，如 C++、Visual C++、Java、C#等。

1.2.2 C 语言的标准

C 语言的灵活性、丰富性、可移植性很快得到了普遍的认可，接着适合于各种操作系统（UNIX、MS-DOS、CP/M-80/86 等）和不同机型（字长为 8 bit～32 bit）的 C 语言编译系统相继出现。1982 年，美国国家标准学会（American National Standards Institute，ANSI）语言标准化委员会开始着手进行 C 语言的标准化工作，并于 1983 年公布了第一个 C 语言标准草案（’83 ANSI C）。1989 年，ANSI 又发布了一个完整的 C 语言标准——ANSI X3.159-1989，通常称为“ANSI C”，简称“C89”。1990 年，国际标准化组织 ISO/JEC JTC1/SC22/WG14 采纳了 C89，做了少量编辑性修改后，以国际标准 ISO/IEC 9899:1990 发布，通常称其为“C90”，它同 C89 基本上相同。

1995 年，WG14 对 C89 做了两处技术修订和一个扩充，人们将其称为“C89 增补 1”或“C95”。同时，WG14 开始着手对 C 标准做全面修订，并于 1999 年获得通过，形成正式的 C 语言标准，

命名为 ISO/IEC 9899:1999，简称“C99”。

本书将基于 C99 介绍 C 语言程序设计的基本方法。目前各厂商提供的所有 C 编译器都尚未实现 C99 所建议的功能。为了使读者能实际运行 C 程序，本书所介绍的程序都是符合 ANSI C 标准并能在大多数 C 编译器上通过和运行的程序。但在文字叙述中，会介绍 C99 所增加的新功能，以使读者今后能够顺利地过渡到用 C99 编写程序。

1.3 C 语言程序概要

1.3.1 函数

任何一部机器都是由部件组装而成的。计算机程序同机器一样，也可以由一些部件构建起来。C 语言程序的部件是函数。也就是说，设计 C 语言程序就是设计其构成函数。

下面举例说明 C 语言程序中的函数是怎样的。

例 1.1 输出字符串的 C 语言程序。

```
/* 文件名: ex010101.c */
# include <stdio.h>

int main(void)
{
    printf(("Programming is fun.");          /* 输出一串字符          */
    return 0;                                /* 向操作系统返回数字 0 */
}
```

这是一个非常简单的 C 语言程序，它的执行结果是显示一行字符：

```
Programming is fun.
```

需要说明以下几点。

（1）这里

```
int main (void)
{
    ...
}
```

是一个函数。这个函数的名字是“main”。这个名字是专用的，表示这个函数是“主函数”。所谓主函数，就是在执行这个程序时，由操作系统直接调用的函数。每一个 C 语言程序必须也只能有一个主函数。

（2）函数名后面的圆括号用于表示参数。一般来说，用函数进行计算，需要提供一些参数。但是，广义的计算也可不含任何参数而只执行一个过程。在 C 语言程序中，参数部分写为“void”（或空）表示该函数没有参数，只执行一个过程。例如，上述程序第一行可写为

```
int main()
```

在许多程序中，常常可以见到这种形式的主函数首行。但是，C 标准建议主函数无参数时写出“void”，以使得含义清晰。本书的程序都按照这一建议的形式书写。

（3）再后面的一对花括号中的部分称为函数体，用来表明该函数的功能如何实现。通常，函数体的主要成分是语句。C 语言规定语句必须以分号结束。先分析下面的语句

```
printf("Programming is fun.");
```

其功能是调用编译系统所提供函数库中的一个函数 printf()来输出后面的字符串。函数 printf 的使用比较复杂，后面将进一步介绍。

（4）函数名前面的“int”表明该函数的返回值是一个整数。有的操作系统（如 UNIX）要求在执行一个程序之后应向系统返回一个整数值，如程序正常执行和结束，应返回 0，否则返回一个非 0 值。因此，需要将 main 函数指定为 int（整型），同时在函数体的最后添加返回语句

```
return 0;
```

其功能是向调用者（操作系统）返回 0 值，表示主函数正常结束（也就是程序正常结束）。此语句必须写在函数体的最后一行才有意义，因为只要执行到这条语句，就表示程序正常结束，向操作系统返回一个 0。如果程序未执行到这条返回语句就非正常地结束，则不会向操作系统返回 0。操作系统会据此做出相应的处理。

有的操作系统（如 DOS、Windows）并无程序必须返回某一整数的要求，因此，可以不指定 main 函数为整型。这时可在 main 函数的前面加上 void，如

```
void main(void)  或  void main()
```

表示 main 函数是无类型的，不返回任何类型的值。显然在 main 函数的最后也不必添加返回语句“return 0;”。读者可以在其他教材或程序中看到这种形式的 main 函数。

以上两种用法都是合法的、有效的，编程人员可以根据具体情况做出决定。为了使程序具有一般性，本书采用以下形式

```
int main(void)
```

并在函数体最后添加“return 0;”语句。

（5）程序最前面的

```
# include <stdio.h>
```

是在程序编译之前要处理的内容，称为编译预处理命令。编译预处理命令还有很多，它们都以“#”开头，.并且不用分号结束，所以不是 C 语言的程序语句。这里的编译预处理命令称为文件包含命令，其作用是在编译之前把程序需要使用的关于系统定义的函数 printf()的一些信息文件 stdio.h 包含进来。以“.h”作为后缀的文件称为头文件。

（6）“/* … */”中的文字是一些说明性文字——注释，用以增加程序可读性，不被编译，也不被执行。例如，注释

```
/* 文件名：ex1_01.c */
```

告诉读程序的人，这个程序的源代码通过文件 ex1_01 保存。而其他两条注释是对其左边两条语句功能的说明。

上面的程序只由一个函数组成(在主函数中又调用了库函数 printf())。在例 1.2 中将介绍由两个函数所组成的程序。

例 1.2 计算两个整数 2、3 相加的结果。

```
/* 文件名：ex010201.c */
# include <stdio.h>

int add (int, int );                    /* 声明将要使用的函数 add() */
```