C Programming Language

C语言程序设计

郝长胜 杜鹏东 编

高等教育出版社
HIGHER EDUCATION PRESS

C 语言程序设计

C Yuyan Chengxu Sheji

郝长胜　杜鹏东　编

高等教育出版社·北京
HIGHER EDUCATION PRESS　BEIJING

内容提要

本书是面向高等学校“C 语言程序设计”课程而编写的教材。全书共 11 章，主要内容包括：C 语言概述，数据类型、运算符及表达式，顺序结构程序设计，选择结构程序设计，循环结构程序设计，数组，指针，函数，预处理，结构体和共用体，文件。从第 3 章开始，大部分章节以一个小案例的形式引入该章节教学内容，采用“案例驱动”的编写风格，以程序设计为主线，内容介绍由浅入深，语法精炼，案例生动易懂，配套使用《C 语言程序设计习题与实验指导》，可以帮助读者进行上机操作，从而提高上机实训能力。

本书可以作为高等学校非计算机专业程序设计类参考教材，也可作为全国计算机 C 语言程序设计二级考试的辅导教材，还可以作为自学 C 语言程序设计人员的参考用书。

图书在版编目（CIP）数据

C 语言程序设计 / 郝长胜，杜鹏东编．— 北京：高等教育出版社，2012.2
ISBN 978-7-04-030629-3

Ⅰ．①C… Ⅱ．①郝… ②杜… 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(2011)第279856号

策划编辑 李 林　　责任编辑 张 珊　　封面设计 张志奇　　版式设计 王 莹
责任校对 杨雪莲　　责任印制 韩 刚

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 北京汇林印务有限公司
开　　本 787 mm×1092 mm 1/16
印　　张 17.25
字　　数 410 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.landraco.com
　　http://www.landraco.com.cn
版　　次 2012 年 2 月第 1 版
印　　次 2012 年 2 月第 1 次印刷
定　　价 28.00 元

物 料 号 30629-00

前　言

程序设计类课程是高等学校非计算机专业必修的计算机基础课程，它以编程语言为平台，重点介绍常用的程序设计思想与方法。通过学习，学生可以掌握高级程序设计语言知识，并且能够在实践中掌握程序设计的基本思想与方法。可以说这是一门以培养学生掌握程序设计的基本方法与提高实践技能为目标的课程。

C语言是目前许多高校首选的程序设计类语言，它不仅具有高级语言所共有的特性，而且可以利用它编写具有操纵计算机硬件能力的程序。它功能丰富，表达能力强大，使用灵活方便，应用面广，且可移植性好，利用它不仅可以编写应用软件，还可以编写系统软件。

本书内容理论联系实际，采用“案例驱动”、“问题引入”的编写风格，每一个案例都以程序点拨、设计思路及思考提示等为主线，清晰的设计格式便于读者学习。本书共11章。第1章C语言概述，主要介绍C语言的发展历程，C语言的结构特点以及C语言的编译、执行过程；第2章数据类型、运算符及表达式，主要介绍C语言程序中所涉及的基本数据类型及定义方法；第3章顺序结构程序设计，主要介绍顺序结构的设计思想，自顶向下的设计理念；第4章选择结构程序设计，主要介绍选择结构类型及其使用方法；第5章循环结构程序设计，主要介绍循环结构类型及其使用方法；第6章数组，主要介绍数组的定义及使用方法；第7章指针，主要介绍指针的定义及使用方法；第8章函数，主要介绍函数定义及使用方法；第9章预处理，主要介绍宏及文件包含的使用方法；第10章结构体与共用体，主要介绍结构体数据类型的定义及使用方法；第11章文件，主要介绍文件的类型及文件常用函数的使用方法，另外，在本书的后面还有附录，主要介绍C语言的关键字、ASCII码、运算符及常用库函数。

本教材获得2011年度内蒙古科技大学教材基金资助，由郝长胜、杜鹏东编，李海荣、刘永花、裴衣非、卢凤、董焕芝、胡晓燕参与编写。其中第1、9章及附录由李海荣编写，第2、6章由郝长胜、刘永花编写，第3~5章由裴衣非编写，第7、11章由杜鹏东、卢凤编写，第8、10章由董焕芝、胡晓燕编写。

与本书配套的还有《C语言程序设计习题与实验指导》，内容包含《C语言程序设计》中的章节习题及解答、Visual C++6.0集成环境介绍、配套实验及考试系统的介绍。

由于编者水平有限，书中难免有不妥之处，望广大读者指正。

编者

2011年10月

目　录

第1章 C语言概述

【学习目标】

(1) 了解计算机语言的发展简史及其分类。
(2) 了解C语言的发展简史及其特点。
(3) 熟悉C语言程序的一般结构。
(4) 掌握C语言程序的编辑、编译、连接和调试的过程。

【学习重点】

C语言程序的编辑、编译、链接和调试的过程。

【学习难点】

(1) C语言的结构。
(2) 程序的编译和链接。

1.1 计算机语言简介

为了交流的方便,人类发明了语言,从此语言就成为人类描述现实世界、表达自己思想感情的工具。因而在不同的地理环境,不同的历史条件下,形成了多种语言,如汉语、英语、法语、德语等。在计算机出现以后,为了能够与计算机"交流",让其按照人类的意图工作,人类发明了计算机语言(又称程序设计语言)。计算机语言是人类与计算机之间的桥梁,一方面人类使用各种计算机语言将所关心的现实世界映射到计算机世界中;另一方面,人类又可以通过计算机语言创造现实世界中并不存在的虚拟世界。

随着计算机技术的发展,各种不同风格的计算机语言不断涌现。但总的来说,可将计算机语言划分为低级语言和高级语言两大类,其中低级语言包括机器语言和汇编语言。

1. 机器语言

在计算机诞生之初,产生了机器语言(Machine Language),这是一种纯粹的二进制语言,用不同的二进制代码来代表不同的指令,计算机可以直接执行这些指令。但是用机器语言编写程序十分困难,编程人员要熟记计算机的全部指令代码和代码的含义。而且编写出的程序全是一些由0和1组成的指令代码,直观性差,且还容易出错。另外,不同类型的计算机有着不同的指令系统,这就使得在某一类计算机上编写的程序不能够在另一类计算机上运行。

现在,除了计算机生产厂家的专业人员外,绝大多数程序员已经不再去学习机器语言了。

2. 汇编语言

为了克服机器语言难读、难编、难记和易出错的缺点，人们就用与代码指令实际含义相近的英文缩写词、字母和数字等符号来取代指令代码（如用 ADD 表示运算符号“+”的机器代码），于是就产生了汇编语言（Assembly Language）。

汇编语言由于采用了助记符号来编写程序，比用机器语言中的二进制代码编程要方便些，在一定程度上简化了编程过程。但是计算机是不认识这些符号的，这就需要一个专门的程序，负责将这些符号翻译成二进制代码的机器语言，这种翻译程序被称为汇编程序。

汇编语言同样十分依赖于计算机硬件，移植性差，但效率高，针对计算机特定硬件而编制的汇编语言程序，能准确发挥计算机硬件的功能和特长，所以至今仍是一种常用而强有力的软件开发工具，尤其是在编制系统软件和过程控制软件方面有着较大的优势。

3. 高级语言

不论是机器语言还是汇编语言都是面向硬件操作的，由于这类语言对计算机过分依赖，要求使用者必须对硬件结构及其工作原理十分熟悉，这对非计算机专业人员是难以做到的，对于计算机的推广应用不利。于是为了使人们从烦琐的指令系统中解脱出来，经过努力，1954 年第一个完全脱离计算机硬件的高级语言——Fortran 问世了。在随后的五十多年中，共有几百种高级语言出现，有重要意义的有几十种，影响较大、使用较普遍的有 Fortran、ALGOL、COBOL、Basic、LISP、Pascal、C、PROLOG、Ada、C++、VC++、VB、Delphi、Java 等。

高级语言接近于人的自然语言并且能为计算机所接受，而且书写出的程序可读性强，易于移植，因此得到人们的青睐。尤其是 C 语言，一直是应用最广泛的语言。从 20 世纪 80 年代初开始，面向对象的设计思想得到人们的广泛认同，它强调的基本原理是直接面对客观事物本身进行抽象并在此基础上进行软件开发，将人类的思维方式与表达方式直接应用在软件设计中，C++、Java、Perl 等语言就是其中的典型代表。

用高级语言编写的源程序，计算机是不能够直接运行的。源程序在输入计算机时，必须通过“翻译程序”翻译成机器语言形式的目标程序，计算机才能识别和执行。这种“翻译”通常有两种方式，即编译方式和解释方式。

编译方式是指在源程序执行之前，就利用编译器（Compiler）将程序源代码“翻译”成目标代码（Object Code，以一个或者几个 *.obj 文件存储），然后通过链接程序将目标程序链接成可执行程序。因为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。C 语言属于编译型的高级语言。

解释方式是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），一边执行该代码，因此效率比较低，而且不能生成独立的可执行文件，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。Basic 语言属于解释型的高级语言。

编译方式和解释方式这两者的区别在于：编译是将源程序翻译成可执行的目标代码，翻译与执行是分开的；而解释是对源程序的翻译与执行一次性完成，不生成可存储的目标代码。

4. 面向对象语言

从 20 世纪 80 年代初开始，在软件设计思想上，又产生了一次革命，其成果是出现了面向对象的程序设计。在此之前的高级语言，几乎都是面向过程的，程序的执行像流水线，在一个模块被执行完成前，不能干别的事，也无法动态地改变程序的执行方向。这和人们日常处理事物的方式

是不一致的，对人而言是希望发生一件事就处理一件事，也就是说，不能面向过程，而是面向具体的应用功能，也就是对象(Object)。其方法就是设计一些通用的、封装紧密的功能模块，它与具体应用无关，但能相互组合，完成具体的应用功能，同时又能重复使用。对使用者来说，只关心它的接口及可实现的功能，至于是如何实现的，则完全不用关心，如 C++、Java 等语言就是典型的代表。

小提示：

1. 一种语言的所谓高级、中级和低级并不是看它的功能强大、先进与否，这里的“高”和“低”是指离硬件的远和近，即低级语言更容易操作计算机的硬件。

2. 移植性是指为某一种计算机编写的软件可适用于另一种类型的计算机。例如，如果为苹果机编写的一个程序能够方便地改为可以在 IBM PC 上运行的程序，则称为是可移植的。

1.2 C 语言的发展及特点

1.2.1 C 语言的发展

20 世纪 60 年代，贝尔实验室(Bell Laboratory)的 Ken Thompson 为了开发 UNIX 操作系统，将 BCPL 语言改造成更适合开发 UNIX 的 B 语言(取 BCPL 的第一个字母)，并用 B 语言编写了 UNIX 操作系统初版。但 B 语言过于简单，功能有限。1972 年，贝尔实验室的 D.M.Ritchie 在 B 语言的基础上设计出 C 语言(取 BCPL 的第二个字母)，它保留了 B 语言的精炼、接近硬件的优点，又克服了其过于简单、无数据类型的缺点。1973 年，Ken Thompson 和 D.M.Ritchie 利用 C 语言重写了 UNIX 操作系统。很快，UNIX 得到广泛的应用，同时 C 语言也迅速地得到推广，成为使用最广泛的语言。但随着微型计算机的日益普及，出现了许多 C 语言版本。由于没有统一的标准，使得这些 C 语言之间出现了一些不一致的地方。为了改变这种情况，1989 年，美国国家标准协会(American National Standard Institute，ANSI)为 C 语言制定了一套 ANSI 标准，成为现行的 C 语言标准。目前流行的各种版本的 C 语言都是以它为基础的，如 Turbo C、Microsoft C、Quick C 等。这些版本的 C 语言虽然基本部分相同，但也有些差异，读者在学习时应注意自己所使用的 C 编译系统的特点和相关规定。本书的叙述是以 ANSI C 为基础，所列程序是在 VC++6.0 环境下调试通过的。

值得一提的是，在 1983 年，Ken Thompson 和 D.M.Ritchie 共同获得了图灵奖。虽然这个奖项的授予是因为他们在 UNIX 上的开创性工作，但谁也不能忽视 C 语言在其中发挥的巨大作用。

1.2.2 C 语言的特点

C 语言发展迅速，成为最受欢迎的语言之一，主要因为它具有强大的功能。许多著名的系统软件，如 Windows、Linux、DBase Ⅲ PLUS、DBASE Ⅳ 都是由 C 语言编写的。归纳一下，C 语言有如下特点。

1. C 语言是高级语言中的中级语言

人们常称 C 语言是一种中级语言，是指 C 语言既具备高级语言使用方便、接近自然语言和

数学语言的特性,同时也具备对计算机硬件系统良好的操纵和控制能力。C语言的这种中级语言特性使得程序在具有较好的可读性和可移植性的同时,又允许程序员能够像用汇编语言编程那样对机器的硬件操纵自如。

2. C语言是结构式语言

结构式语言的显著特点是代码及数据的分隔化,即程序的各个部分除了必要的信息交流外是彼此独立的。这种结构化方式可使程序层次清晰,便于使用、维护和调试,也有利于大型软件的协作开发。而且C语言是以函数形式提供给用户的,这些函数可以方便地调用,并具有多种循环、条件语句来控制程序的流向,从而使程序完全结构化。

3. 语言简洁紧凑

C语言一共有32个关键字(见附录A),9种控制语句,程序书写自由,而且运算符、语句等语言成分的表示简明扼要,为程序员减少了编程需要记忆的内容,有利于人们对编程语言的学习与使用。

4. 运算符丰富

C语言的运算符包含的范围很广泛,共有34个运算符,表达式类型多样化。灵活使用各种运算符可以实现在其它高级语言中难以实现的运算。

5. 数据结构丰富

C语言的数据类型有整型、实型、字符型、数组类型、指针类型、结构体类型和联合体类型等。它们能用来实现各种复杂的数据类型的运算,并引入了指针概念,使程序效率更高。另外C语言具有强大的图形功能,支持多种显示器和驱动器,并且计算功能、逻辑判断功能也比较强大。

6. 效率高

C语言程序经编译后生成的目标程序代码质量高,或者说效率高。代码质量是通过C语言源程序经编译后生成的目标程序在运行速度和存储空间上的开销大小来衡量的。运行速度越快,占用的存储空间越少,则代码质量越高。C语言在代码质量上几乎可以与汇编语言相媲美。

7. 适用范围大,可移植性好

用C语言编写的程序基本上不用修改就能适用于各种型号的计算机及各种操作系统。

C语言的这些特点使得C语言既适合用于编写系统软件(如操作系统、编译系统等),也适合编写应用软件(如图形处理、信息处理等),因此成为最为流行的程序设计语言之一。

任何事物的好和坏都是相对的,C语言也有自己的不足之处。C语言与其它高级语言相比,有碎、繁、难的特点。另外由于C语言放宽了语法检查,增大了程序编写的灵活性,同时也增大了出错的概率,在一定程度上降低了系统的安全性。这就需要程序员依靠自身的能力来提高程序的安全性和可靠性。

由于网络和多媒体等技术发展很快,计算机操作系统平台也在不断更新。目前C语言逐渐偏居于一隅,取而代之的是C++、Java和C#等程序语言。在这种情况下,为什么还要学习C语言呢?其实C语言可以说是C++、Java和C#语言的基础,而且很多专用语言也学习和借鉴C语言,如进行Web开发的PHP语言,做仿真的MATLAB的内嵌语言等。因此,学好C语言对以后学习其它语言大有帮助。在日新月异的计算机时代,各种语言"你方唱罢我登台",唯有掌握最基础的语言,才能以不变应万变,立于不败之地。

1.3 C 语言的基本结构

下面通过几个简单的例子，来了解 C 语言的基本结构，以便使大家对 C 语言有个感性的认识。

学习某种编程语言，通常采用程序 Hello World（如同一个生命面对世界宣告自我存在的第一声呼唤）作为起步，下面就开始编写第一个 C 程序—— “Hello World!”。

【例 1-1】 在屏幕上输出 Hello World!

【源程序】

```
/*This is a C program.*/
# include <stdio.h>
void main( )  // 在屏幕上输出 Hello World!
{
    printf("Hello World!\n"); // 函数体
}
```

这是一个非常简单的 C 语言程序，它的功能就是在屏幕上输出 “Hello World!”。

【运行结果】

如图 1.1 所示。

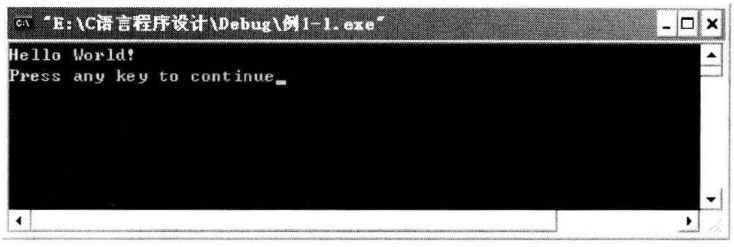

图 1.1 例 1-1 运行结果

对例 1-1 作如下说明。

(1) 程序的第 1 行是注释语句。

为了增加程序的可读性和易懂性，C 语言程序中可以加入注释进行说明。注释对程序的运行无任何作用，即程序对注释部分不予执行，注释的多少也不影响程序。

在 C 程序中，注释有两种方式。

一是多行注释，注释由 “/*” 开始，由 “*/” 结束，在 “/*” 和 “*/” 之间写入注释的内容。

二是单行注释，在 “//” 后加入注释信息（如第 4 行后面部分）。

(2) 程序的第 2 行是文件包含预处理命令。

C 语言的预处理命令都是以 “#” 开头，其中 stdio.h 是 C 语言系统文件，stdio 是标准输入 / 输出的缩写，.h 是文件的扩展名，它表明该文件是一个头文件（head file）。通过文件包含命令 #include 把输入 / 输出头文件 stdio.h 的文件内容读到该命令的位置处。例 1-1 包含头文件的原因是在程序中要调用 C 语言中的库函数中的格式输出函数 printf( )。有关预处理命令，将在第 9

章中进行介绍。

(3) 程序的第3行是空行。

C语言程序允许插入空行,它不影响程序的功能。必要的空行可增加程序的可读性。

(4) 程序的第4~7行是程序的主函数。

main是主函数名,一个C语言程序有且仅有一个主函数,并且程序执行时从主函数的"{"开始,在主函数的"}"结束。main前的"void"(空)表示这个主函数不返回值。花括号{ }中间是main( )函数的具体实施部分,称为函数体。printf( )是一个C语言提供的库函数,它的作用是在屏幕上输出指定的内容。

(5) 函数体中的每一条语句都用";"表示语句的结束。

需要指出的是,#include是编译预处理命令,而不是语句,所以不以";"结束,并且要求单独占一行。

【例1-2】 用自定义函数的方法计算两个整数的和。

【源程序】

```
/*This is a C program. */
#include <stdio.h>
void main( )
{                                   /* 程序开始 */
    int sum(int a,int b);          /* 函数原型说明 */
    int x,y,s;                     /* 声明变量 */
    x=20;y=40;
    s=sum(x,y); /* 调用 sum( )函数计算 x 和 y 之和 */
    printf("The sum is %d\n",s); /* 输出结果 */
}                                   /* 程序结束 */

int sum(int a,int b)
{
    int c;                          /*sum( )函数的声明部分 */
    c=a+b;
    return(c); /* 将 c 的值返回,通过 sum( )函数带回调用处 */
}
```

主函数首部

主函数体

主函数

自定义函数 sum( )首部

函数体

自定义函数

【运行结果】

如图1.2所示。

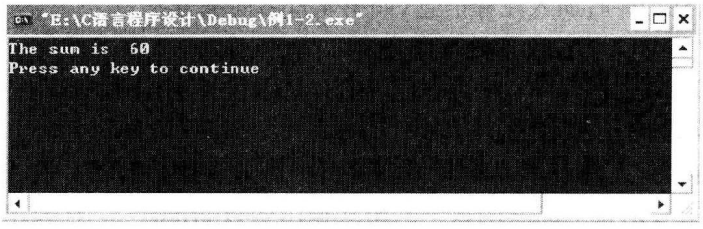

图1.2 例1-2运行结果

对例 1-2 作如下说明。

(1) 程序定义了两个函数,主函数 main( )和自定义函数 sum( )。

自定义函数的结构与主函数的结构类似,都分成两部分,即首部(即函数的第一行)和函数体。函数体由大括号{ }括起来,函数体由声明部分和执行部分组成。声明部分定义变量,是对函数中“对象”的描述;执行部分代表操作,是对函数所要实现的“动作”的描述。在 C 程序中,函数的声明部分一定要在执行部分之前,它们的顺序不能颠倒。

另外,根据需要,一个 C 程序可以包含 0 到多个用户自定义函数。因此,一般 C 程序的结构如下所示,其中 f1,f2,…是自定义函数,其函数体由一对大括号{ }括起来。

```
预处理命令和全局性的声明。
main( )
{
    主函数体
}
f1( )
{
    函数体
}
f2( )
{
    函数体
}
…
```

由此可见,函数是 C 语言的基本单位,即 C 程序是由函数构成的。函数既可以是系统提供的库函数(如 printf( )函数),也可以是用户根据需要自己编写的函数(如例 1-2 中的 sum( )函数)。C 语言的函数库十分丰富,ANSI C 提供了 100 多个库函数,Turbo C 和 MS C 4.0 提供了 300 多个库函数。

(2) 本程序中主函数调用了自定义函数。

在程序中,语句“s=sum(x,y);”实现了对 sum( )函数的调用。调用时,将实际参数 x 和 y 的值分别传给 sum( )函数中的形式参数 a 和 b,然后开始执行 sum( )函数中的语句,当执行语句“return(c);”后,结束 sum( )函数的执行,得到一个返回值 c,并将这个值赋给 s,然后将其输出。

(3) C 语言程序的书写格式。

C 程序书写格式自由,一行内可以写多条语句,一条语句也可以分别写在多行上。

例 1-1 的程序可以写成:

```
void main( )
{
   printf(
"Hello World ! "
);
```

```
}
```

但是必须注意,在一条C程序语句中,并不是所有的部分都可以拆开写在多行上,语句中的有些部分作为整体只能写在同一行上而不允许将其拆开。如输出函数printf( )中,用双撇号括起来的内容是作为函数的一个独立参数存在,它作为一个整体就不能被拆开,如以下写法就是错误的。

```
void main( )
{
    printf("Hello
World! ");
}
```

建议初学者每行写一条语句,而每行的语句根据需要适当向右缩进几列,这对读懂程序和调试程序很有帮助。本书中的所有程序都按最常用的格式书写。

为了说明C语言程序的结构,在例1-2中用到了函数调用、形式参数、实际参数等概念,如果读者对此不太理解,也不必深究,在以后的学习中,还会详细介绍。介绍这个例子的目的是要使大家对C语言程序的组成和形式有一个初步的了解。

到目前为止,只是完成了程序的书写,并不代表编程工作就此结束,那么C语言编程到底需要几步呢?

1.4 C语言程序的运行

C语言是一种编译型程序语言。运行一个C语言程序一般需要4个步骤:

上机输入、编辑源程序 → 对源程序进行编译 → 与库函数链接 → 运行目标程序。此过程如图1.3所示。

1. 编辑

编辑(Edit)是C语言程序开发的第一步,工作内容是输入、修改程序。通常使用的编译程序都是集成化的,开发一个C语言程序的所有工作,都可以通过它完成。通过编辑得到的程序称为源程序(源文件),源程序以纯文本格式保存在源程序文件(简称为程序文件)中,源文件名的后缀一般要求为.c,如file1.c。

2. 编译

对源程序的语法和逻辑结构等进行检查以生成目标文件(Object)的过程就是编译(Compile/Make)。如果源程序存在语法错误,用户应该根据错误提示信息查找错误并改正,然后重新编译,直到没有语法错误为止。编译后生成目标文件,目标文件与相应的源程序文件的主文件名相同,但扩展名为.obj,如file1.obj。

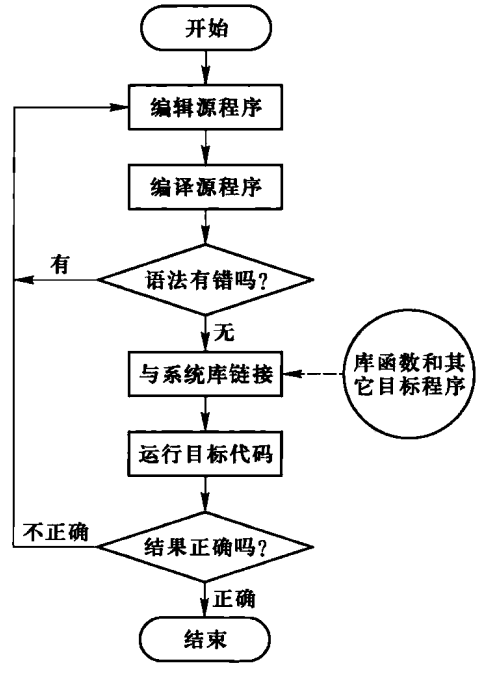

图1.3 C语言程序的运行步骤

3. 链接

源文件经过编译后产生的目标文件还不能直接运行。链接(Link)的作用是把目标文件、其它目标程序模块与系统提供的标准库函数有机地结合起来,生成可以运行的可执行文件。可执行文件名与相应的目标文件及源文件的主文件名相同,但扩展名为 .exe,如 file1.exe。 当程序调用了不存在的函数或函数名时,就会出现链接错误,这时同样需要重新修改源程序。

4. 执行

链接成功后得到的是一个可执行(Run)文件,可执行文件可以直接去执行应用。可执行文件生成后便和源代码脱离关系,它的执行并不需要源代码的存在。 如果可执行文件运行后,得到正确的结果,表示程序正确;如果得到的结果不符合要求,则表示程序存在逻辑错误,这种情况需要重新修改源程序。

可以总结出在整个程序运行过程中生成的程序文件如图 1.4 所示。

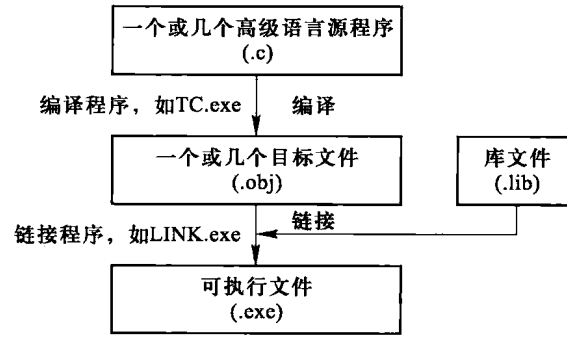

图 1.4 程序运行过程中生成的文件

小提示:

C 程序的错误可以分为两种:语法错误和逻辑错误。

① 语法错误:这是 C 语言初学者出现最多的错误,比如,分号";"是每个 C 语句的结束的标志,在 C 语句后忘记写";"就是语法错误,发生语法错误的程序,编译通不过,可以根据开发环境的提示信息来修改。

② 逻辑错误:就是用户编写的程序已经没有语法错误,可以运行,但得不到所期望的结果(或正确的结果),也就是说由于程序设计者的原因,程序并没有按照程序设计者的思路来运行。比如一个最简单例子是:本想求两个数的和,应该写成 z=x+y;由于某种原因却写成了 z=x-y;这就是逻辑错误。编译软件发现不了逻辑错误,要用户跟踪程序的运行过程才能发现,这是最不容易修改的。

1.5 C 语言的学习建议

作为初学者,学习 C 语言很容易进入两个误区。

第一,纯粹学语法。学习语法是必要而且重要的,但如果把学习的重点只是放在语法的理解和掌握上,那就本末倒置了。就好像学英文,一个只懂语法而不会说英文的人算不算学好了英文呢?答案当然是否定的。所以应该明确,学语言的目的是为了写程序,学语法是为了帮助人们更好地写程序。

第二,只看不写。学习 C 语言要非常重视上机实验,不上机做实验是不可能学好 C 语言的。程序是写会的,不是看会的。

那么,到底该怎么才能学好 C 语言呢? 下面是给大家一些建议,供大家参考。

第一,C语言是一门实践性很强的课程,除了要像其它课程一样,做好预习、复习工作外,注重每一次的上机实验尤为重要。

第二,要注意养成良好的程序书写风格。包括结构层次缩进风格、符号书写风格、必要的注释等。好的书写风格可以使程序易于阅读、扩充、修改和维护。

第三,注意写好程序的每一句。在写每一句代码时想一想:这句为什么要这样写? 为什么要放在这里,可以不要吗?

第四,经常读别人编的好程序,如书上的例题等,要注意模仿。要注意体会别人设计程序的用心和思想,想一下,这个程序如果自己要编,会怎么去做? 可以把自己当成CPU,在大脑中一步步运行程序。

第五,初学C语言时,可能会遇到有些问题理解不透,这就要求对不明白的地方多问多想,鼓足勇气进行学习,待学完后面的章节知识,前面的问题也就迎刃而解了,就是说,学习后面的知识,不要忘记回头弄清遗留下的问题和加深理解前面的知识,这是非常重要的。

总之,希望大家在本书的指导和帮助下能够学好C语言,让它成为学习和工作的好帮手。

本章小结

本章主要介绍了有关计算机语言的基本知识、C程序的结构和上机操作步骤。

计算机系统是由硬件和软件两部分组成的。计算机软件最主要的内容就是程序。通俗地讲,程序就是让计算机解决问题的方法步骤。这种方法步骤必须用计算机可以识别的符号描述出来,也就是用某种计算机语言进行程序设计。

程序设计的步骤一般要经过分析问题和解决问题两大步骤。一个好程序应当具有正确性、易读性、健壮性和可移植性等特点。

要验证一个程序是否正确就要通过程序调试,其步骤包括编辑、编译、链接和运行。目前C语言比较流行的编译系统是Turbo C 2.0和Visual C++6.0,这些都是集成化的编译工具,上述几个步骤都可以在这些环境中实现。