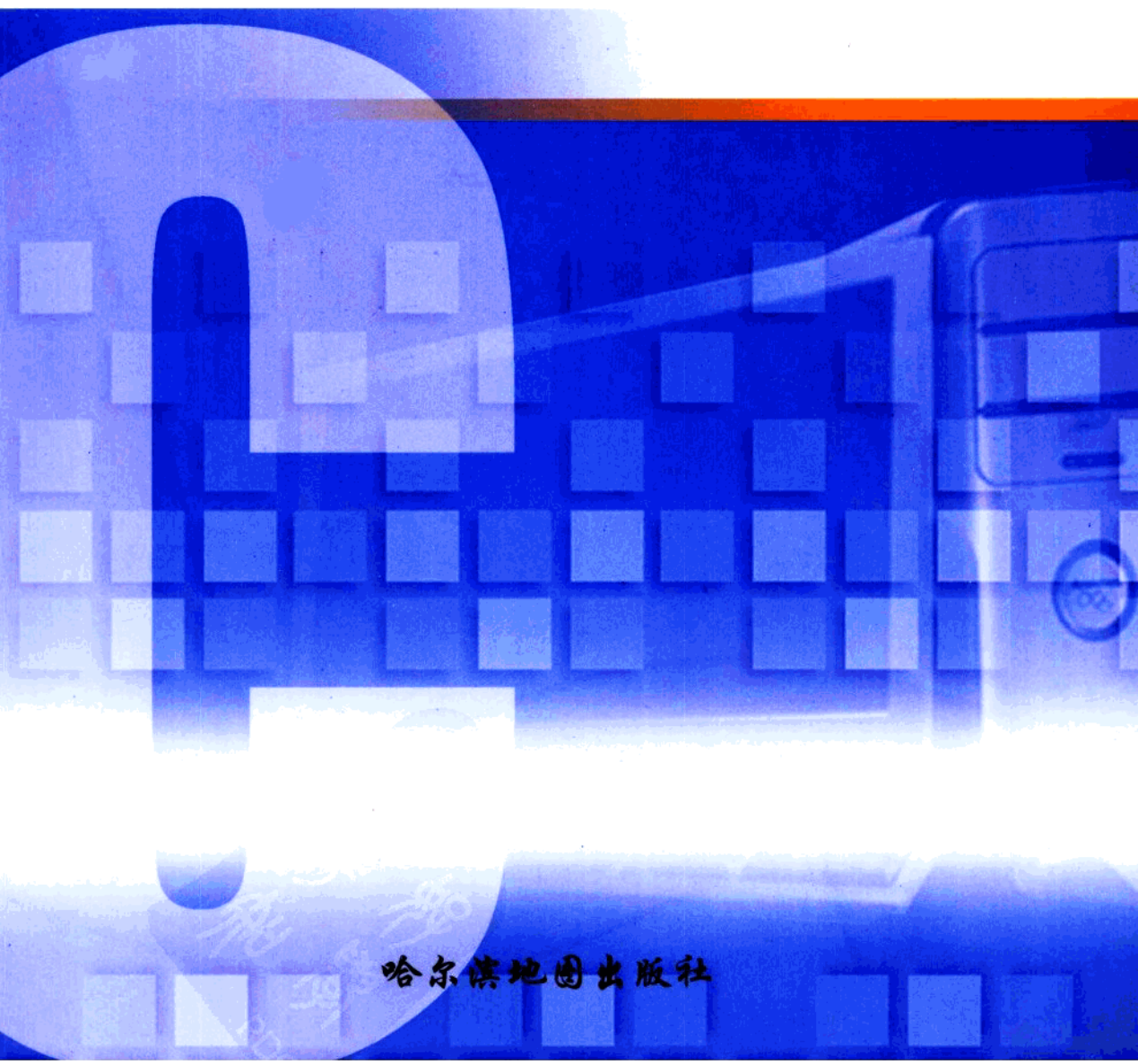
C语言程序设计教程

主　编　刘向阳　王春霞

副主编　任竞颖　崔卫东　王海燕



哈尔滨地图出版社

前　言

C 语言程序设计是目前各本科院校和高等职业技术学校计算机专业学生的专业必修课，也是计算机爱好者自学程序设计语言的首选对象。即使是一些理工科非计算机专业的学生，也有很多人把 C 语言作为一门重要的课程来学习。对于参加计算机等级考试（二级）的学生，大部分也把 C 语言作为了选考内容。如何帮助学生更好地掌握程序设计的基本思想，熟练地运用 C 语言去设计实用软件，能在较好地掌握 C 语言的基础上，进一步地学习和掌握更高级的面向对象的程序设计语言（如 Visual C++，JAVA 等），如何能够提高学生编程解决实际问题的能力，是各院校计算机专业的教师共同关心的问题。

本书依据高等院校面向 21 世纪 C 语言教学大纲，结合编者多年教学的实践经验编写了该书。本书的主要特点是：突出设计思想的分析讲解，力求使学生掌握程序设计的思路和常用的解题算法。围绕此中心，组织了一些有代表性的例题和练习。特别是在每章的最后都配有实验，以加深学生对本章节内容的理解和掌握。同时，本书力求概念清楚，通俗易懂。

本书由刘向阳策划、统稿，由刘向阳、王春霞任主编，任竞颖、崔卫东、王海燕任副主编。参编人员还有王迤冉、李杰、叶斌、田素霞、李英等。

在本书的策划和编写过程中，得到了商丘师范学院计算机科学系张庆政主任、管典兵主任、陈树平教授的指导和大力支持，在此向他们表示诚挚的感谢。

由于编者水平有限，加之编写时间仓促，书中一定存在不少的缺点和错误，恳请读者批评指正。

编　者

2008 年 4 月

目　录

第 1 章　概述

1.1　计算机语言概述

计算机的核心部件是 CPU，计算机完成的每一种操作都是在 CPU 的控制下完成的。CPU 需要执行由人们输入的指令，才能完成一定的任务。一般情况下，CPU 能完成一系列操作，每种操作对应一条或几条指令。我们把计算机能执行的指令的集合称作指令系统。实际上，这些指令就是人们同计算机“交流的语言”。计算机所完成的任务往往需要若干条指令组合起来，形成人们称之为 “程序”的指令的有序集合。编制程序这一过程就是程序设计。设计中使用的语言就是计算机语言。

1.1.1　计算机语言的发展概况

当代计算机在程序控制下能完成相当复杂的任务，但计算机内部所能识别的只有“0”、“1”两个二进制数。也就是说，指令在计算机内部实际上是由 0 和 1 组合起来形成的二进制编码。由二进制编码形成的计算机语言称为机器语言。使用机器语言编程，要求编程者精通计算机的内部结构，熟记由 0 和 1 编码形成的指令，程序设计相当困难，影响了计算机的普及与应用。

为了解决用计算机编程的难题，人们研制出助记符来代替机器语言编程，这种由助记符表示的计算机语言为汇编语言。使用汇编语言编程，克服了机器语言难记、不易阅读等缺点。但汇编语言同机器语言一样，也是面向特定计算机的，要求编程者精通计算机的内部结构，通用性差，只适用于专业开发人员使用。

为了普及计算机的应用，让计算机发挥更大的作用，人们研究开发出更易掌握、接近人类自然语言的计算机语言，这就是高级语言。用高级语言开发应用程序较机器语言和汇编语言要容易，效率也高。用汇编语言或高级语言编写的程序，一般称为汇编语言或高级语言源程序。除了机器语言外，使用其他语言编写的源程序，在计算机内部都必须转换成机器语言。源程序都可以通过被称为编译或解释系统的“翻译”来实现这种转换。编译和解释系统是由专业开发人员设计的系统程序，不同的高级语言有它自己的编译或解释系统。

早期使用的高级语言种类很多，如 FORTRAN 语言、BASIC 语言、ALGOL 等，这些语言是面向数学公式和算法的语言。20 世纪 80 年代，计算机语言跨入了面向对象的编程阶段，出现了 C++，Perl 等与自然语言更接近的计算机语言。20 世纪 90 年代，出现了网络化的编程语言，如 JAVA 语言。计算机语言的发展，丰富了计算机的功能。如今，无论在生活领域，还是在工业控制等领域，计算机已经显示出不可替代的作用。

1.1.2 C语言的发展历史

C 语言是 1972 年由美国的 Dennis Ritchie 设计发明的，并首次在 UNIX 操作系统的 DEC PDP-11 计算机上使用。它由早期的编程语言 BCPL（Basic Combined Programming Language）发展演变而来。在 1970 年，AT&T 贝尔实验室的 Ken Thompson 根据 BCPL 语言设计出较先进的并取名为 B 的语言，并用 B 语言写了第一个 UNIX 操作系统。1972 年，贝尔实验室的 Ritchie 在 B 语言的基础上设计出了 C 语言，取的是 BCPL 语言的第二个字母。到了 1973 年，两人又用 C 语言一起改写了 UNIX 操作系统 90%以上的代码，也就是 UNIX 第 5 版本。1978 年，Ritchie 和 Brain Kernighan 编写了《The C Programming Language》，并于 1988 年作了修订，该书就是 C 语言版本的基础，被称为标准 C。随着计算机的日益普及，出现了许多 C 语言版本，正是 C 语言具有的可移植性的特点，使得 UNIX 系统在 AT&T, VAX 等计算机系统上能够迅速实现。但是由于没有统一的标准，使得这些 C 语言之间出现了不一致的地方。为了改变这种情况，美国国家标准化协会（ANSI）为 C 语言制定了一套 ANSI 标准，成为现行的 C 语言标准，包括了 1983 年和 1987 年两个版本。到了 1990 年，国际标准化组织（International Standard Organization，简称 ISO）接受了 1987 年 ANSI C 作为了 ISO C 的标准，即 ISO 9899－1990。目前比较流行的 C 语言编译系统都是以此为基础，如 Turbo C，Borland C，Microsoft C 等。

1.1.3 C语言的特点

C 语言发展如此迅速，并且成为最受欢迎的语言之一，主要因为它具有强大的功能。C 语言是从“组合编程语言”CPL 发展而来，既具有一般高级语言特性(ALGOL 60 带来的高级语言特性)，又具有低级语言特性(BCPL 带来的接近硬件的低级语言特性)。

C 语言的特点有：

(1) C 语言提供的语句简洁，使用方便，格式紧凑，语法灵活。C 语言一共有 32 个关键字，9 种控制语句，语句简练，书写自由。以下关键字是由系统定义的，不能用做其他定义。

与数据类型相关的关键字：

```
char        int          short         long          signed       unsigned
float    double    enum       struct      union      typedef     void
```

与存储类型相关的关键字：

```
auto          register       static       extern
```

与控制语句相关的关键字：

```
if        else          switch          case          default        do
while        for                break           continue              goto
return
```

其他的关键字：

```
const          sizeof         volatile
```

下面是 C 语言中的 9 种控制语句：

```
if(…)…else…        switch(…) …case        for(…) …       while(…) …
do…while(…)        continue          break          goto          return
```

（2） C 语言的运算符十分丰富，一共有 34 种运算符，有算术、关系、逻辑、位、赋值、指针、条件、逗号、下标、类型转换运算符等多种类型。

（3） 数据结构多样，有整型、实型、字符型、枚举类型等基本类型，有数组、结构体、共用体等构造类型以及指针类型，还为用户提供了自定义数据类型，特别是引入了指针概念，能够实现复杂的数据结构。

（4） C 语言的控制语句形式多样，使用方便。有两路分支、多路分支和循环结构几种控制语句，以及结构化模块的实现和控制，便于程序的编制和维护。

（5） C 语言是一种模块化的程序设计语言，采用自顶向下、逐步求精的结构化程序设计方法，各模块功能独立，以函数形式编制，通过函数之间的相互调用和数据传递，实现系统整体的功能要求。这样，把大型系统的实现化整为零，便于分工合作以及共享。

例 1.1 第一个 C 程序。

```
/*这是我的第一个 C 程序*/                    /*  注释  */
#include <stdio.h>                          /*  编译预处理  */
void main( )                                /*  主函数  */
{                                           /*  函数开始  */
printf("Welcome to C world!");              /*  语句  */
}                                           /*  函数结束  */
```

其中，第一行用/* */括起来的表示注释，第二行是编译预处理，第三行是主函数名，void 表示主函数没有返回值，第四、六行的一对大括号界定函数体，表示函数的开始和结束，第五行是一条 C 语句。

例 1.2 从键盘上输入两个整数进行比较，并输出其中的较小值。

```
#include <stdio.h>
void main( )
{
  int a, b, result;                         /*  变量声明  */
  scanf("%d,%d", &a, &b);                   /*  接收两个整数的输入  */
  if (a<b) result = a;                      /*  判断两个数的大小  */
      else result = b;
  printf("The min number is: %d", result); /*  输出结果  */
  getchar( );                               /*  运行后等待用户按键返回  */
}
```

例 1.3 通过函数调用实现求两个数的较小值。

```
#include <stdio.h>
void main( )
{
  int a, b, result;
  scanf("%d,%d", &a, &b);
  result = min(a, b);                       /*  调用函数  */
  printf("The minimum number is: %d", result);
  getchar( );
```

```
}
int min(int x, int y)                    /*  函数首部  */
{                                        /*  函数开始  */
        int num;                         /*  函数内部变量声明  */
        if (x<y) num = x;
        else num = y;
        return(num);                         /*  函数值返回  */
}                                            /*  函数结束  */
```

通过上面的三个简单的实例可以看出，C 程序是由一个或多个函数组成的，但是有且仅有一个主函数 main。程序的执行是从 main 函数开始，在 main 函数中结束，其他函数通过调用得以执行。

C 程序中，所有的关键字都是小写字母，在 C 程序里大小写是敏感的，一般习惯使用小写字母作为变量名称和字符串常量。C 语言书写格式自由，不使用行号，可以使用空格和空行，但习惯上采用上面例子中的锯齿形书写格式。良好的书写习惯，是优秀的程序员必备的素质之一。例如，使用“Tab”键缩进，大括号对齐，在适当的地方添加空行，并添加必要的注释。

（6） C 语言可以直接访问地址、进行位运算，从而对硬件进行操作，因此 C 语言既具有高级语言编写简单方便、便于理解的优点，又具有低级语言与硬件结合紧密的优点。因此 C 语言被称为介于高级语言和低级语言之间的中级语言。

（7） C 语言具有很强的移植性，由 C 语言编写的程序基本不用太多的修改就可以用于不同型号的计算机上，程序和硬件的匹配由 C 语言的编译程序完成，同时也可以在多种操作系统下使用。

（8） C 语言具有很好的通用性，既可以用于编写应用软件，也适合编写系统软件。例如 UNIX 操作系统的源代码就是用 C 语言编制的。

1.2 程序设计基础知识

什么是程序语言?什么是程序设计？应该学哪一种程序语言?如何进行程序设计？这些都是程序设计初学者首先遇到的问题，也是程序设计的基本问题、共性问题。不论是什么样的计算机语言，其程序设计的基本方法是相同的。本书作为程序设计的入门教材，将以C语言程序设计为主线，介绍程序设计的基本概念和基本方法，讲述C语言的语法规则和实用的C程序设计技术。

1.2.1 程序与程序语言

（1）计算机语言

什么是计算机语言？为什么要使用计算机语言？过去，一提到语言这个词，人们自然想到的是像英语、汉语等这样的自然语言，因为它是人和人相互交流信息不可缺少的工具。而今天，计算机遍布我们生活的每一个角落。除了人和人之间的相互交流之外，我们必须和计算机交流。用什么样的方式和计算机做最直接的交流呢？人们自然想到的是最古老也

最方便的方式——语言。人和人交流用的是双方都能听懂和读懂的自然语言，同样，人和计算机交流也要用人和计算机都容易接受和理解的语言，这就是计算机语言。人们用自然语言讲述和书写，目的是给他人传播信息。同样，我们使用计算机语言把我们的意图表达给计算机，目的是使用计算机。

计算机语言是根据计算机的特点而编制的，它没有自然语言那么丰富多样，只是有限规则的集合，所以它简单易学。但是，也正因为它是根据机器的特点编制的，所以交流中无法意会和言传，而更多地表现了说一不二，表现了“规则”的严谨。例如该写“；”的地方不能写成“.”，该写“a”的地方不能写成“A”，这使得人和计算机的交流在一开始会有些不习惯。不过，只要认识到计算机语言的特点，注意学习方法，把必须的严谨和恰当的灵活相结合，一切都会得心应手。

（2）程序

计算机是一种具有内部存储能力的自动、高效的电子设备，它最本质的使命就是执行指令所规定的操作。如果我们需要计算机完成什么工作，只要将其步骤用诸条指令的形式描述出来，并把这些指令存放在计算机的内部存储器中，需要结果时就向计算机发出一个简单的命令，计算机就会自动逐条顺序执行操作，全部指令执行完就得到了预期的结果。这种可以被连续执行的条条指令的集合称为计算机的程序。也就是说，程序是计算机指令的序列，编制程序的工作就是为计算机安排指令序列。

但是，我们知道，指令是二进制编码，用它编制程序既难记忆，又难掌握。所以，计算机工作者就研制出了各种计算机能够懂得、人们又方便使用的计算机语言，程序就是用计算机语言来编写的。因此，计算机语言通常被称为“程序语言”，一个计算机程序总是用某种程序语言书写的。

（3）程序语言的发展

程序语言的产生和发展，直接推动了计算机的普及和应用。自第一个高级语言问世以来，人们已发明了上千种程序语言，常用的也有上百种。这些语言之间有什么区别，我们应该学习哪一种？

计算机语言按使用方式和功能可分为低级语言和高级语言。低级语言包括机器语言和汇编语言。机器语言就是计算机指令的集合，它与计算机同时诞生，是第一代的计算机语言；汇编语言是用符号来表示计算机指令，被称为第二代语言。机器语言和汇编语言都是围绕特定的计算机或计算机族而设计的，是面向计算机的语言。要使用这种语言必须了解计算机的内部结构，而且难学、难写、难记忆，把这种语言称为低级语言。因为低级语言是难以普及应用的，为此便产生了第三代语言——高级语言。它采用了完全符号化的描述形式，用类似自然语言的形式描述对问题的处理过程，用数学表达式的形式描述对数据的计算过程。可见，高级语言只是要求人们向计算机描述问题的求解过程，而不关心计算机的内部结构，所以把高级语言称为“面向过程语言”，它易于被人们理解和接受。典型的面向过程语言有 BASIC，FORTRAN，COBOL，C，PASCAL，等等。

随着计算机技术的迅猛发展，20 世纪 80 年代以来，众多的第四代非过程化语言、第五代智能化语言也竞相推出。如果说第三代语言要求人们告诉计算机怎么做，那么第四代语言只要求人们告诉计算机做什么。因此，人们称第四代语言是“面向对象语言”。面向对象概念的提出是相对于“面向过程”的一次革命，面向对象技术在系统程序设计、多媒体应用、数据库等诸多领域得到广泛应用。但是，“面向过程”是程序设计的基础，尤其

对程序设计的初学者。所以，我们将以面向过程的C程序设计语言为背景，主要介绍程序设计的基本概念和方法。

（4）程序设计及程序设计语言

程序是用程序设计语言表示的计算机解题算法或计算机解题任务。程序设计是将解题任务转变成程序的过程，一般包括：分析问题；确定算法（对复杂算法需画出流程图）；用选定的程序设计语言编写源程序；上机调试、运行程序等基本步骤。程序设计语言是计算机能够理解的、用于人和计算机通信的语言，程序设计语言可分为低级语言、高级语言、面向对象的语言和专用语言四类。

①低级语言

低级语言又分为机器语言、符号语言和汇编语言。机器语言用二进制代码表示机器指令和数据，机器语言程序能够直接被机器理解和执行，虽然这种语言程序效率高，但编程烦琐，且不便于记忆和阅读，因而程序维护困难。符号语言用符号代替二进制代码表示机器指令，而汇编语言用符号来表示指令和数据的内存地址。现在人们用低级语言编程通常指用汇编语言编程。

用汇编语言编写的程序称为汇编语言源程序。汇编语言程序必须被转换为机器语言程序才能被计算机理解和执行，完成这种转换任务的系统软件称为汇编程序，这种转换过程称为汇编。低级语言是面向机器的，用低级语言写的程序效率高，但没有可移植性，即不能从一个机器系统上移到另一个机器系统上运行。此外，用机器语言编写程序要求程序员必须懂得具体机器系统的硬件结构（指令系统）。

②高级语言

高级语言是类似于人类自然语言的英语和数学语言的程序设计语言，例如 C 语言、PASCAL 语言。用高级语言编写的程序称为高级语言源程序，简称源程序或程序。例如C源程序或C程序，PASCAL 源程序或 PASCAL 程序。

与汇编语言源程序类似，高级语言源程序也必须被转换成机器语言程序才能够被机器理解和执行，完成这种转换任务的系统软件称为编译程序。例如C编译程序、PASCAL 编译程序。将高级语言源程序转换成机器语言程序的过程称为编译。

高级语言是面向解题过程的，语言本身与具体机器系统无关，因而用高级语言编写的应用程序可移植性好。编译程序是一种语言的具体实现，编译程序与具体机器系统有关，人们常将一个编译程序称为某种语言的一个版本。同一种语言的不同版本不完全相同，在使用一种具体的高级语言及其编译程序开发软件时，必须参考与编译程序配套的有关资料。本书阐述的内容严格遵循C语言标准，对于大多数C编译程序具有通用性。考虑到上机环境尽量简单以便于学习和练习，书中所有例题均在 Turbo C 2.0 上通过运行。

③面向对象的语言

典型的、目前应用最普遍的面向对象的语言是C++语言，C++语言是20世纪80年代为适应中型和大型复杂软件的开发和维护而发展起来的面向对象的程序设计语言，其目标是为程序员的程序开发活动提供一个优良的程序设计环境，以产生模块化程度高、可重用性和可维护性好的程序。20世纪90年代推出的JAVA语言是一种适合于分布式计算的新型面向对象的程序设计语言，JAVA语言主要用于 Internet 应用程序的开发和编制。

④专用语言

专用语言是专门为特定的应用领域而设计的计算机程序设计语言。例如，计算机辅助

设计(CAD)系统中使用的绘图语言，数据库管理系统(DBMS)中的数据查询语言 SQL、军事应用领域的 ADA 语言、商业应用领域的 COBOL 语言等。专用语言是面向问题的语言，它比高级语言更抽象，描述能力比高级语言更强。用专用语言编程不需指出“如何做”，只需说明“做什么”。

1.2.2 程序设计

什么是程序设计呢？在日常生活中我们可以看到，同一台计算机有时可以画图，有时可以制表，有时可以玩游戏，诸如此类。也就是说，尽管计算机本身只是一种现代化方式批量生产出来的通用机器，但是，使用不同的程序，计算机就可以处理不同的问题。今天，计算机之所以能够产生如此之大的影响，其原因不仅在于人们发明了机器本身，更重要的是人们为计算机开发出了不计其数的能够指挥计算机完成各种各样工作的程序。正是这些功能丰富的程序给了计算机无尽的生命力，它们正是程序设计工作的结晶。而程序设计就是用某种程序语言编写这些程序的过程。

更确切地说，所谓程序，是用计算机语言对所要解决问题中的数据以及处理问题的方法和步骤所做的完整而准确的描述，这个描述的过程就称为程序设计。对数据的描述就是指明数据结构形式。

对于程序设计的初学者来说，首先要学会设计一个正确的程序。一个正确的程序，通常包括两个含义：一是书写正确，二是结果正确。书写正确是指程序在语法上正确，符合程序语言的规则；而结果正确通常是指对应于正确的输入，程序能产生所期望的输出，符合使用者对程序功能的要求。程序设计的基本目标是编制出正确的程序。但这仅仅是程序设计的最低要求。一个优秀的程序员，除了注重程序的正确性外，更要注重程序的高质量。所谓高质量是指程序具有结构化程度高、可读性好、可靠性高、便于调试维护等一系列特点。毫无疑问，无论是一个正确的程序，还是一个高质量的程序，都需要设计才能使之达到预期的目标。那么，如何进行程序设计呢？一个简单的程序设计一般包括以下四个步骤：

（1）分析问题，建立数学模型

使用计算机解决具体问题时，首先要对问题进行充分的分析，确定问题是什么，解决问题的步骤又是什么。针对所要解决的问题，找出已知的数据和条件，确定所需的输入、处理及输出对象。将问题过程归纳为一系列的数学表达式，建立各种量之间的关系，即建立起解决问题的数学模型。需要注意的是，有许多问题的数学模型是显然的或者简单的，以至于我们没有感觉到需要模型。但是有更多的问题需要靠分析来构造计算模型，模型的好与坏、对与错，在很大程度上决定了程序的正确性和复杂程度。

（2）确定数据结构和算法

根据建立的数学模型，对指定的输入数据和预期的输出结果，确定存放数据的数据结构。针对所建立的数学模型和确定的数据结构，选择合适的算法加以实现。注意，这里所说的“算法”泛指解决某问题的方法和步骤，而不仅仅是指“计算”。

（3）编制程序

根据确定的数据结构和算法，用自己所使用的程序语言把这个解决方案严格地描述出来，也就是编写出程序代码。

（4）调试程序

在计算机上用实际的输入数据对编好的程序进行调试，分析所得到的运行结果，进行

程序的测试和调整，直至获得预期的结果。

由此可见，一个完整的程序要涉及四个方面的问题：数据结构、算法、编程语言和程序设计方法。这四个方面的知识都是程序设计人员所必须具备的，其中算法是至关重要的一个方面。关于数据结构和算法问题有专门的著作。本书的重点是介绍编程语言和程序设计方法。但是，如果我们对算法还一无所知，就无法进行基本的程序设计。

1.3 C 语言程序

1.3.1 C 语言程序

（1） C语言程序的特点

1）C 语言程序的构成

C 语言的源程序存放在扩展名为.C 的文件中，源程序是由函数构成的，至少包含一个 main（） 函数，或者 main（） 函数和其他函数。函数是 C 程序的基本单位，可以由用户根据自己的需要进行定义，也可以是系统提供的库函数。

由用户编写的 main（） 函数和若干其他函数从结构和功能上都相对独立，只是由于相互之间的调用关系而联系在一起。C 程序的执行从 main（） 函数开始，再通过 main（） 函数对其他函数的调用以及其他函数相互之间的调用实现各函数功能，然后层层返回各级调用点，最后返回到调用的起点 main（） 函数，从而结束一次程序的运行。

2）函数的组成

一个函数是由函数首部和函数体组成的。函数首部包括函数的返回值类型、函数名、函数的形式参数类型和名称，函数名后的圆括号内是形式参数表，如果有多个参数则用逗号隔开，也可以没有参数，则括号内为空；函数体用一对大括号括起，包括变量的定义和由语句组成的执行部分，由用户根据函数的功能进行编写，是实现函数功能的主体。定义函数的格式如下：

<函数返回值类型>　　<函数名>(<形式参数表>)

{

变量的定义；　语句序列；

}

函数的开始和结束标志就是指包括函数体的最外层的一对大括号。函数体的执行从左大括号后的第一条语句开始，到右大括号前的最后一条语句结束。在函数体中，由于复合语句的存在，可能出现若干个或若干层大括号，要分清楚对应关系，不要漏掉。最外层的大括号才是函数的开始和结束标志。

①main（） 函数。main（） 函数是每个 C 语言源程序的必然组成部分。无论 main（） 函数在文件的哪个位置，C 程序的执行总是从 main（） 函数开始，最后回到 main（） 函数结束。main（） 函数可以去调用其他函数，但不能被其他函数所调用。

②其他函数。除 main（） 函数之外的其他函数，有的由用户编写，有的是系统库函数。它们可以被 main（） 函数调用，但不能去调用 main（） 函数，其他函数相互之间

也可以相互调用。标准 C 提供了一百多个库函数，Turbo C 2.0 和 Microsoft C 则提供了四百多个函数。用户自定义函数采用上面的格式编写，主要根据功能进行设计，同时考虑与其他函数的调用关系。

③函数之间的调用。函数的调用从 main（） 函数开始，先由 main（） 函数调用其他函数，然后其他函数再发生以下若干级的调用，再根据调用顺序层层返回调用点，最后返回到 main（）函数结束。如图 1.1 所示，简单勾画出函数之间的调用关系，让大家有一个初步的印象。

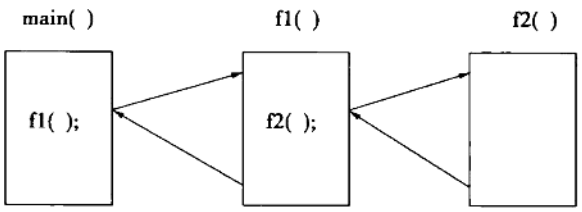


图 1.1 C程序的函数调用

3）源程序的书写格式

①C 语言的书写格式比较自由，一行可以写几条语句，一个语句也可以写在多行上。但每条语句和数据定义后必须有一个分号，一行中的多条语句之间也要用分号相隔。

②C 语言的一般语句和函数以及数据定义最好用小写字母表示，而符号常量名用大写字母表示，以便于区分。

③C 语言的注释符号由/*和*/组成，之间的字符即为注释信息，可以独成一行，也可以跟在某一行语句的后面，注释信息用于对程序或语句的功能进行解释说明，起到增强程序可读性的作用。

（2）C语言基本语法成分

1)C语言字符集

字符是 C 语言的最基本的元素，C 语言字符集由字母、数字、空白、标点和特殊字符组成（在字符串常量和注释中还可以使用汉字等其他图形符号）。由字符集中的字符可以构成C语言进一步的语法成分（如：标识符，关键词，运算符等）。

①字母：A～Z，a～z

②数字：0～9

③空白符：空格、制表符（跳格）、换行符（空行）的总称。空白符除了在字符、字符串中有意义外，编译系统忽略其他位置的空白。空白符在程序中只是起到间隔作用。在程序的恰当位置使用空白将使程序更加清晰，增强程序的可读性。

④标点符号、特殊字符：

!	#	%	^	&	+	−	*	/	=	~	<	>	\
¦	.	,	;	:	?	‘	“	(	)	[	]	{	}

2）标识符（名字）

用来标识变量名、符号常量名、函数名、数组名、类型名等实体(程序对象)的有效字