

全国高等职业教育规划教材

C 语言程序设计

芦秀莲 ◎ 主 编

 中央广播电视大学出版社

全国高等职业教育规划教材

C 语言程序设计

芦秀莲 主 编

中央广播电视大学出版社

北 京

图书在版编目 (CIP) 数据

C 语言程序设计 / 芦秀莲主编. —北京: 中央广播电视大学出版社, 2011.10

全国高等职业教育规划教材

ISBN 978-7-304-05259-1

I. ①C… II. ①芦… III. ①C 语言—程序设计—
高等职业教育—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字 (2011) 第 197880 号

版权所有, 翻印必究。

全国高等职业教育规划教材

C 语言程序设计

芦秀莲 主编

出版·发行: 中央广播电视大学出版社

电话: 营销中心: 010-58840200 总编室: 010-68182524

网址: <http://www.crtvup.com.cn>

地址: 北京市海淀区西四环中路 45 号

邮编: 100039

经销: 新华书店北京发行所

策划编辑: 苏 醒

印刷: 北京雷杰印刷有限公司

版本: 2011 年 11 月第 1 版

开本: 787×1092 1/16

责任编辑: 冯 欢

印数: 0001~3000

2011 年 11 月第 1 次印刷

印张: 18 字数: 271 千字

书号: ISBN 978-7-304-05259-1

定价: 33.00 元

(如有缺页或倒装, 本社负责退换)

编写人员

主 编：芦秀莲

编 委：（以姓氏笔画为序）

王 溧 韦月稳 吕秋旋 刘建宏

孙 丽 李 燕 李富英 李相君

吴一心 陈 晨 邹 敏 赵世华

郭 勤 秦义宾 曹玉斌 彭 博

覃琼花 覃飞云 蒋振宇 曾少志

蔡洪亮 戴湘黔

内容提要

C语言是目前最流行的程序设计语言之一，具有简洁、紧凑、灵活、实用、高效、可移植性好等优点，深受广大用户欢迎。C语言的数据类型丰富，既可以用来编写系统程序，又可以用来编写应用程序。因此，C语言迅速被推广和普及。

本书通过大量的实例，从计算机语言和程序设计的基本知识、C语言的发展与特点出发，系统地介绍了C语言程序设计中的变量、运算符、表达式、数据类型、存储类别、语句、函数、指针、位运算和文件等。因此本书将以面向过程的C语言为背景，介绍程序设计的基本概念和基本方法。在本书最后一章，简单介绍了面向对象程序设计的基本方法及流行的面向对象程序设计语言C++和C#。同时，每章都配有精选的练习题，作为对该章内容的巩固和延伸。本书结构简洁明快、重点突出、通俗易懂、逻辑性强；始终以程序设计为主线，注重培养读者程序设计的思维方式和技巧。

本书可作为高职高专高等院校开设“C语言程序设计”课程的教材，也可作为成人教育以及在职人员的培训教材，同时也可供全国计算机等级考试者参考。

目 录

第1章 概述

1.1 程序设计语言	1
1.1.1 计算机语言	1
1.1.2 程序设计语言的发展	1
1.2 C 语言程序设计	2
1.2.1 C 语言的发展史	2
1.2.2 C 语言的主要特点	3
1.3 C 程序的基本结构	3
1.3.1 C 程序的实例	3
1.3.2 输入/输出函数的简单介绍	5
1.3.3 C 程序的基本结构	6
1.3.4 C 语言的词汇	6
1.3.5 C 程序的书写	8
1.4 C 程序的开发过程	9
本章习题	11

第2章 基本数据类型、运算符及表达式

2.1 C 语言的数据类型	13
2.1.1 数据类型	13
2.1.2 C 语言数据类型简介	13
2.1.3 C 语言的基本数据类型	14
2.1.4 基本数据类型的修饰	14
2.2 常量	15
2.2.1 整型常量	15
2.2.2 实型常量	16
2.2.3 符号常量	17
2.2.4 字符型常量	18
2.2.5 字符串常量	20
2.3 变量	21
2.3.1 定义变量	21
2.3.2 变量类型	21
2.4 运算符与表达式	24
2.4.1 C 语言运算符简介	24

2.4.2 表达式	25
2.4.3 算术运算符和算术表达式	26
2.4.4 赋值运算符和赋值表达式	29
2.4.5 逗号运算符和逗号表达式	30
2.4.6 条件运算符和条件表达式	30
2.4.7 关系运算符和关系表达式	32
2.4.8 逻辑运算符和逻辑表达式	34
2.4.9 数据之间的混合运算	36
本章习题	37

第 3 章 结构化程序设计

3.1 结构化程序设计的思想及流程图	42
3.1.1 结构化程序设计思想	42
3.1.2 流程图	42
3.2 结构化程序设计的 3 种基本结构	43
3.2.1 顺序结构	43
3.2.2 选择结构	44
3.2.3 循环结构	56
3.2.4 循环的嵌套	61
3.3 几种转移控制语句	62
3.3.1 break 语句	62
3.3.2 continue 语句	63
3.3.3 goto 语句	65
3.3.4 return 语句	65
3.4 数据的输入与输出	65
3.4.1 不同数据类型的输入	66
3.4.2 不同数据类型的输出	70
本章习题	75

第 4 章 数组

4.1 一维数组	80
4.1.1 一维数组的定义	80
4.1.2 一维数组的初始化	81
4.1.3 一维数组的引用	82
4.1.4 一维数组应用举例	82
4.2 二维数组	85
4.2.1 二维数组的定义	85
4.2.2 二维数组的初始化	86
4.2.3 二维数组的引用	88

4.3 字符数组	89
4.3.1 字符数组的定义和初始化	89
4.3.2 字符串与字符数组	91
4.3.3 字符串处理函数	94
4.3.4 字符数组应用举例	97
本章习题	100

第5章 函数

5.1 C 程序的模块结构	106
5.1.1 C 程序的模块化结构概念	106
5.1.2 函数的概述	106
5.2 函数的分类	107
5.3 函数的定义	108
5.3.1 函数的说明	108
5.3.2 函数的定义	109
5.3.3 有参函数和无参函数	111
5.4 函数的调用	112
5.4.1 函数调用的一般形式	112
5.4.2 函数调用的方式	113
5.5 函数间的参数传递	114
5.5.1 函数的参数和值	114
5.5.2 用数组作为函数参数进行传递	116
5.6 函数的嵌套	117
5.7 函数的递归	117
5.8 变量的作用域	121
5.8.1 局部变量	121
5.8.2 全局变量	122
5.9 变量的存储类型	123
5.9.1 静态存储方式和动态存储方式	124
5.9.2 静态局部变量	124
5.9.3 自动变量	125
5.9.4 寄存器变量	126
5.9.5 静态全局变量和非静态全局变量	127
5.9.6 变量在使用中的注意事项	128
5.10 系统函数	130
5.10.1 常用的数学函数	130
5.10.2 字符函数	133
5.10.3 其他函数	135
本章习题	136

第6章 结构体与共用体

6.1	结构体类型的定义	142
6.1.1	结构体类型的定义说明	142
6.1.2	定义结构体类型时需要注意的问题	143
6.1.3	结构体型的嵌套定义	143
6.2	结构体变量的定义及内存分配	144
6.2.1	结构体变量的定义	144
6.2.2	结构体类型的内存分配模式	146
6.3	结构体变量的初始化和引用	146
6.3.1	结构体变量的初始化	146
6.3.2	结构体变量的引用	148
6.4	结构体数组	149
6.4.1	结构体数组的定义	149
6.4.2	结构体数组的初始化	150
6.5	结构体变量和结构体数组成员的访问	151
6.5.1	访问结构体成员运算符	151
6.5.2	访问结构体成员的方法	152
6.6	指向结构体类型数据的指针	154
6.6.1	结构体指针的初始化	154
6.6.2	结构体指针的应用	154
6.7	结构体型与函数	156
6.7.1	结构体型的数据作函数的参数	156
6.7.2	结构体型的函数	160
6.8	用递归结构处理链表	163
6.8.1	递归结构	163
6.8.2	链表的概念	164
6.8.3	动态链表的基本操作	164
6.9	共用体型	167
6.9.1	共用体的定义	167
6.9.2	共用体变量的定义	168
6.9.3	共用体变量的引用	168
6.9.4	共用体类型数据的特点	171
6.9.5	结构体和共用体的区别	172
6.10	枚举类型	173
6.10.1	枚举型的定义	173
6.10.2	枚举变量的定义	173
6.10.3	给枚举常量赋初值	174
6.10.4	枚举中的几条规定	174

6.11 用 typedef 定义类型	176
6.11.1 typedef 定义类型的格式	176
6.11.2 typedef 定义类型的几点说明	177
6.11.3 类型定义的使用	177
本章习题	179

第 7 章 指针

7.1 地址和指针的概念	186
7.1.1 变量的内容和变量的地址	186
7.1.2 直接访问和间接访问	187
7.1.3 指针的概念	188
7.2 指针变量	188
7.2.1 指针变量的定义	188
7.2.2 指针变量赋值	188
7.3 指针变量的运算	189
7.3.1 指针运算符	189
7.3.2 指针变量的运算	191
7.4 指向数组元素的指针	195
7.5 指向多维数组的指针变量	198
7.6 指向字符串的指针变量	200
7.7 指针型函数	203
7.7.1 指针型函数的定义方法	204
7.7.2 函数指针变量	204
7.8 二级指针	206
7.8.1 一级指针和二级指针的概念	206
7.8.2 二级指针的定义	207
7.8.3 二级指针的使用	207
7.9 用指针进行内存动态分配	208
7.9.1 内存动态分配的含义	208
7.9.2 内存动态分配的步骤	209
7.9.3 常用的内存动态分配函数	209
本章习题	211

第 8 章 编译预处理

8.1 文件包含处理——#include	215
8.1.1 文件包含的格式	215
8.1.2 文件包含的功能	216
8.2 宏定义——#define	217

8.2.1 不带参数的宏定义	217
8.2.2 带参数的宏定义	219
8.3 条件编译	220
8.3.1 条件编译命令的形式	220
8.3.2 条件编译的功能	222
本章习题	222
第 9 章 位运算	
9.1 位运算与位运算符	227
9.2 位段	229
9.2.1 位段结构类型及位段结构变量的定义	229
9.2.2 位段结构的存储	231
9.2.3 位段结构的使用	231
9.3 位运算举例	231
本章习题	233
第 10 章 文件	
10.1 文件的概述	236
10.1.1 数据文件	236
10.1.2 文件的存取方式	237
10.1.3 流和文件	237
10.1.4 文件指针类型	238
10.1.5 文件操作的步骤	238
10.2 文件的打开与关闭	239
10.2.1 文件的打开	239
10.2.2 文件的关闭	241
10.3 文件测试函数	242
10.4 文件的读写操作	242
10.4.1 字符读写函数	243
10.4.2 数据读写函数	246
10.4.3 字符串读写函数	248
10.4.4 格式化读写函数	249
10.4.5 其他读写函数	251
10.5 文件的随机读写操作	251
10.6 出错的检测	253
本章习题	254
附录	258

第1章 概 述

1.1 程序设计语言

1.1.1 计算机语言

计算机语言是人们描述计算过程（即程序）的规范书写语言。程序是对计算机处理对象和计算规则的描述。语言的基础是一组记号和语法规则。根据语法规则由记号构成记号串的全体就是语言。

人类是使用像英语、汉语这样的自然语言相互交流和表达思想的。人和计算机的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。计算机语言是根据计算机的特点而编制的，是计算机能够“理解”的语言。它是有限规则的集合。计算机语言不像自然语言那样包含复杂的语义和语境，而是用语法来表达程序员的思想，所以编写程序时必须严格遵守语法规则。

1.1.2 程序设计语言的发展

计算机是一种具有内部存储能力、由程序自动控制的电子设备。人们将需要计算机做的工作写成一定形式的指令，并把它们存储在计算机的内部存储器中。当人们需要结果时就向计算机发出一条简单的命令，计算机就按指令顺序自动进行操作。人们把这种可以连续执行的一条条指令的集合称为程序。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。

程序设计语言伴随着计算机技术的发展层出不穷，从机器语言到高级语言，从面向过程的语言到面向对象的语言。至目前为止，计算机语言的发展大致经历了5代。

第一代语言：机器语言，它将计算机指令中的操作码和操作数均以二进制代码形式表示，是计算机能直接识别和执行的语言。它在形式上是由“0”和“1”构成的一串二进制代码。每种计算机都有自己的一套机器指令。机器语言的优点是无需翻译、占用内存少、执行速度快；缺点是随机而异，通用性差，并且因指令和数据都是二进制代码形式，难于阅读和记忆，编码工作量大，难以维护。

第二代语言：汇编语言，是用助记符号来表示机器指令的符号语言。例如，用 ADD 表示加法，用 SUB 表示减法，用变量取代各类地址，这样构成的计算机符号语言称为汇编语言。用汇编语言编写的程序称为汇编语言源程序。这种程序必须经过翻译（称为汇编），变成机器语言程序才能被计算机识别和执行。汇编语言在一定程度上克服了机器语言难于

辨认和记忆的缺点，但对于大多数用户来说，仍然不便于理解和使用。

第三代语言：高级语言，也称为面向过程的语言。高级语言是具有国际标准的，描述形式接近自然语言的计算机语言。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了对数据的计算过程。常用的计算机高级语言有 Basic、Fortran、Cobol、Pascal 和 C 语言等。由于高级语言只要求人们关心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的语言。使用面向过程的语言编程时，编程者的主要精力放在算法过程的设计和描述上。

第四代语言：非过程化语言，是一种功能更强的高级语言。其主要特点是：非过程性，采用图形窗口和人机对话形式，基于数据库和“面向对象”技术，易编程、易理解、易使用、易维护。但是，程序的运行效率和语言的灵活性不如过程化语言高。常用的非过程化语言有 Visual Basic、Java、C++ 和 Delphi 等。

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”。

第五代语言：智能化语言，主要使用在人工智能领域，帮助人们编写、推理和演绎程序。

目前，国内外大多数计算机上运行的程序，大多是用第三代或第四代计算机语言编写的。因此，应当熟练地掌握用高级语言编写程序的方法和技巧。

1.2 C 语言程序设计

1.2.1 C 语言的发展史

C 语言是一种面向过程的程序设计语言，其特点是：简单灵活的编程语言、丰富的数据结构和操作符、先进的结构化程序控制、良好的程序可移植性和高效率的目标代码。

C 语言的祖先是 ALGOL60。1963 年，英国的剑桥大学和伦敦大学首先将 ALGOL60 发展成 CPL，1967 年，英国剑桥大学的 Martin Richards 将 CPL 改写成 BCPL，1970 年，美国贝尔实验室的 Ken Thompson 将 BCPL 修改成 B 语言，并用 B 语言开发了第一个高级语言的 UNIX 操作系统。1972 年，Ken Thompson 与在 UNIX 系统上的亲密合作者 Dennis Ritchie 将 B 语言改成了 C 语言。1978 年，Brain W. Kernighan、Ken Thompson 与 Dennis Ritchie 合著了著名的《C 语言编程》(*The C Programming Language*)，该书介绍的 C 语言成为后来广泛使用的 C 语言版本的基础。C 语言由于自身的优点，在其后的十几年中得到了广泛的使用，适用于不同机种和不同操作系统的 C 编译系统相继问世。1983 年，美国国家标准局 (ANSI) 制定了 C 语言标准。这个标准不断完善，并从 1987 年开始实施 ANSI 的标准 C。1988 年，ANSI 公布了标准 ANSI C。

C 语言发展迅速，而且成为最受欢迎的语言之一。许多著名的系统软件，如 UNIX 操作系统等都是由 C 语言编写而成的。由于 C 语言的强大功能和各方面的优点逐渐被人们认

识,到了20世纪80年代,C语言开始进入其他操作系统,并很快在各类大、中、小型计算机上得到了广泛的使用,成为当代最优秀的程序设计语言之一。

1.2.2 C语言的主要特点

一种语言之所以能存在和发展,并具有生命力,总是有其特点的。归纳起来,C语言主要有以下一些特点:

(1) C语言简洁、紧凑,使用方便、灵活。C语言一共只有32个关键字,9种控制语句,而且程序书写形式自由,主要用小写字母表示,压缩了一切不必要的成分。

(2) C语言是中级语言,同时具备了高级语言和低级语言的特征。它把高级语言的基本结构和语句与低级语言的实用性结合起来。C语言可以像汇编语言一样对位、字节和地址进行操作,而这三者是计算机最基本的工作单元。换句话说,C语言具有汇编语言的强大功能,却没有汇编语言的难度,特别适合做底层开发。既可以用C语言来设计芯片,也可以用来编写操作系统,如UNIX和Linux就是用C语言编写出来的。

(3) C语言是结构化语言,具有结构化的控制语句。结构化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,彼此独立。这种结构化方式可使程序层次清晰,便于使用、维护以及调试。C语言是以函数形式提供给用户的。这些函数可方便地调用,并具有多种循环、条件语句,控制程序流向,从而使程序完全结构化。

(4) C语言具有各种各样的数据类型。C语言支持各种高级语言普遍使用的基本数据类型,并允许用比基本数据类型构造复杂的数据类型。同时,引入了指针概念,可使程序效率更高。此外,C语言具有强大的图形功能,支持多种显示器和驱动器,而且计算功能、逻辑判断功能也比较强大,可以实现决策目的。

(5) C语言适用范围大。C语言的一个突出的优点就是适合于多种操作系统,如DOS、UNIX,也适用于多种机型。

(6) C语言生成目标代码质量高,程序执行效率高。C语言程序生成的目标代码一般只比汇编程序生成的目标代码效率低10%~20%。

上面只介绍了C语言最容易理解的一般特点,C语言内部的其他特点在后面各章将会作以详细介绍。

1.3 C程序的基本结构

C程序的基本结构是指一个C程序的基本组成部分。本节将通过程序实例来说明C程序的基本结构。

1.3.1 C程序的实例

首先通过一个简单的C程序实例来说明C语言源程序结构的特点和书写格式。

【例 1.1】

```
void main()
{
    printf("Hello, world! \n"); /*输出 Hello Human! */
}
```

这个程序的功能是输出下面一行信息：

Hello, world!

上述程序中：

(1) `main` 是主函数的函数名，表示它是一个主函数。每一个 C 源程序都必须有且只有一个主函数 `main`。

(2) 函数体由大括号 `{}` 括起来。上面例子中的程序体只有一个 `printf` 输出语句。`printf` 是 C 语言中的输出函数，其功能是把要输出的内容送到显示器中进行显示。语句中的双引号用来显示一个字符串，双引号内的字符串将按原样输出；“`\n`”是换行符，即在输出“Hello, world!”后回车换行。

下面再看一个相对复杂的 C 程序。

【例 1.2】

```
#include<math.h> /* include 为文件包含命令 */
#include<stdio.h>
main() /* 主函数 */
{
    double x, y; /*定义变量 */
    printf("input number: \n"); /*输出字符串 "inputnumber:" */
    scanf("%lf", &x); /* 输入变量 x 的值 */
    y=cos(x); /* 求 x 的余弦，并把它赋给变量 y */
    /*显示程序运算结果，即显示 x 和 y 的值 */
    printf("cosine of %lf is %lf\n", x, y);
}
```

这个程序的功能是求从键盘上输入一个数 `x` 的余弦值，然后输出结果。

上述程序中：

(1) `/*……*/`表示注释部分，为便于理解，通常用汉字表示注释，当然也可以用英文或汉语拼音作注释。注释只是给人看的，对编译和运行不起作用。注释可以加在程序中的任何位置。

(2) 在 `main()`之前的两行语句称为预处理命令。关于预处理命令在后面的章节中会详细讲解。预处理命令还有其他几种，这里的 `include` 称为文件包含命令，其意义是把尖括号 `<>`或引号 `""`内指定的文件包含到本程序来，成为本程序的一部分。被包含的文件通常是由系统提供的，其扩展名为 `.h`，因此也称为头文件或首部文件。

(3) 在本例中，使用了三个库函数：输入函数 `scanf`、余弦函数 `cos` 和输出函数 `printf`。`cos` 函数是数学函数，其头文件为 `math.h` 文件，因此，在程序的主函数前用 `include` 命令包含了 `math.h`。`scanf` 和 `printf` 是标准输入/输出函数，其头文件为 `stdio.h`，在主函数前也用 `include` 命令包含了 `stdio.h` 文件。C 语言的头文件中包括了各个标准库函数的函数原型。因此，凡是在程序中调用一个库函数时，都必须包含该函数原型所在的头文件。

**提示**

C 语言规定对 `scanf` 和 `printf` 这两个函数可以省去对其头文件的包含命令。所以在本例中也可以删去第二行的包含命令：`#include<stdio.h>`。在例 1.1 中使用的 `printf` 函数，就省略了包含命令 `#include<stdio.h>`。

(4) 在例题中的主函数体又分为两部分：说明部分和执行部分。

① 说明部分完成变量的类型说明。C 语言规定，源程序中所有用到的变量都必须先说明、后使用，否则将会出错。这一点是编译型高级程序设计语言的一个特点。例 1.1 中未使用任何变量，因此无说明部分。说明部分是 C 源程序结构中很重要的组成部分。例 1.2 中使用了两个变量 `x`, `y`，用来表示输入的自变量和 `cos` 函数值。由于 `cos` 函数要求这两个量必须是双精度浮点型，故用类型说明符 `double` 来说明这两个变量。说明部分后的四行语句为执行部分或称为执行语句部分，用以完成程序的功能。

② 执行部分的第一行是输出语句，调用 `printf` 函数在显示器上输出提示字符串，提示用户输入自变量 `x` 的值。第二行为输入语句，调用 `scanf` 函数，接受键盘上输入的数并存入变量 `x` 中。第三行是调用 `cos` 函数并把函数值送到变量 `y` 中。第四行是用 `printf` 函数输出变量 `y` 的值，即 `x` 的余弦值。到此程序全部结束。

1.3.2 输入/输出函数的简单介绍

输入是将原始数据通过输入设备送入计算机，输出是将保存在内存中的计算结果送到输出设备上。为完成此操作，C 编译系统提供了输入/输出函数。其中，常用的是格式输入函数 `scanf` 和格式输出函数 `printf`。为了方便以后的学习，本节对这两个函数作一简单的介绍，详细的介绍见后面的章节内容。

1. 格式输入函数 scanf

格式输入函数 `scanf` 的功能是按指定的格式输入数据，其一般的调用格式为

```
scanf("格式控制字符串", 参数表);
```

其中，`printf` 是函数名，其后括号中的内容为该函数的参数：格式控制字符串用双引号括起来，用来规定输入格式，其用法和 `printf` 函数中规定的相同；参数表中至少包含一个输入项，且必须是变量的地址，多个输入项之间用逗号隔开。例如，语句 `scanf("%d%d", &a, &b);`；用来接收从键盘输入的两个十进制整数，并分别存放在变量 `a` 和 `b` 中。

2. 格式输出函数 printf

格式输出函数 `printf` 的功能是按指定的格式输出数据，其一般的调用格式为

```
printf("格式控制字符串", 参数表);
```

其中，`printf` 是函数名，其后括号中的内容为该函数的参数：格式控制字符串用双引号括起来，用来规定输出格式，如 `%f` 用来输出实数，`%c` 用来输出字符；参数表中包含零个或多个输出项，这些输出项可以是实数、变量或表达式，多个输出项之间用逗号隔开。例如，语句 `printf("%d, %d", a, b);`；用来按十进制整数形式输出变量 `a` 和 `b`。

**提示**

变量地址的表示形式是在变量名前加上一个“&”。

1.3.3 C 程序的基本结构

函数是 C 程序的基本结构，一个 C 程序由一个或多个函数组成，一个 C 函数由若干条 C 语句构成，一条 C 语句由若干个基本单词组成。

C 函数是完成某个整体功能的最小单位，是相对独立的模块。简单的 C 程序可能只有一个主函数，复杂的 C 程序则可能包含一个主函数和任意多个其他函数。所有 C 函数的结构都包括三部分：函数名、形式参数和函数体。如图 1-1 所示为 C 程序的一般格式。

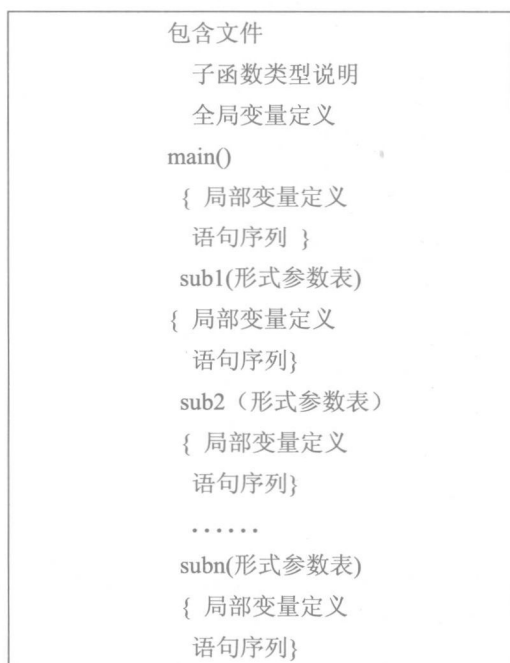


图 1-1 C 程序的一般格式

其中，main 为主函数名，sub1()到 subn()为子函数名。在 C 程序中，主函数名字是固定的，其他的函数名则可以根据标识符的命名方法任意取名；形式参数是函数调用时进行数据传递的主要途径，当形式参数表中有多个参数数，相互之间用逗号隔开。有的函数可能没有形式参数；花括号{}扩起来的部分为函数体，用来描述函数的功能，一般函数体由局部变量定义和完成本函数功能的语句序列组成。程序在执行时，无论各个函数的书写位置如何，总是先执行 main 函数，再由 main 函数调用其他函数，最终终止于 main 函数。

1.3.4 C 语言的词汇

在 C 语言中使用的词汇分为 6 类：标识符、关键字、运算符、分隔符、常量、注释符。