

GENERAL
EDUCATION

高等学校通识教育系列教材

C语言程序设计

冯志红 主编
王春嫻 副主编



清华大学出版社

GENERAL EDUCATION

高等学校通识教育系列教材

C语言程序设计

冯志红 主编

王春嫻 副主编

褚益清 李凤荣 马菲 编著

常海燕 刘晶 刘洋 宁安良

清华大学出版社

北京

内 容 简 介

本书注重培养学生的逻辑思维能力和程序设计能力，是集知识性、实用性及趣味性为一体的 C 语言程序设计教材。

全书共分为 12 章，包括 C 程序设计概述，数据类型、运算符与表达式，顺序结构、选择结构和循环结构程序设计，数组，函数，编译预处理，指针，结构体、共用体和枚举类型，文件，位运算。本书由从事多年 C 程序设计教学的一线教师根据实践教学经验编写而成，内容全面，层次结构清晰，重点突出，逻辑严密，语言通俗易懂，例题丰富；每章后面均有习题供读者练习并附有部分习题答案供参考。

本书适合作为高等学校 C 语言程序设计课程教材，也可作为广大计算机爱好者自学和参考用书。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。

版权所有，侵权必究。侵权举报电话：010-62782989 13701121933

图书在版编目 (CIP) 数据

C 语言程序设计/冯志红主编. —北京：清华大学出版社，2017

(高等学校通识教育系列教材)

ISBN 978-7-302-47273-5

I. ①C… II. ①冯… 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 (2017) 第 125978 号

责任编辑：刘向威

封面设计：文 静

责任校对：焦丽丽

责任印制：王静怡

出版发行：清华大学出版社

网 址：<http://www.tup.com.cn>, <http://www.wqbook.com>

地 址：北京清华大学学研大厦 A 座 邮 编：100084

社 总 机：010-62770175 邮 购：010-62786544

投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈：010-62772015, zhiliang@tup.tsinghua.edu.cn

印 装 者：北京鑫海金澳胶印有限公司

经 销：全国新华书店

开 本：185mm×260mm

印 张：19.75

字 数：478 千字

版 次：2017 年 8 月第 1 版

印 次：2017 年 8 月第 1 次印刷

印 数：1~2000

定 价：45.00 元

前 言

C 语言是目前使用非常广泛的高级程序设计语言。在对操作系统以及硬件进行操作的场合, C 语言明显优于其他高级语言, 许多大型应用软件都是用 C 语言编写的。此外, C 语言绘图能力强, 可移植性好, 并具备很强的数据处理能力。因此, C 语言受到广大计算机专业人员、非专业人员的青睐。

本书以培养学生的逻辑思维能力和实践应用能力为出发点, 从大量实例入手, 采用通俗易懂的语言由浅入深地对 C 语言程序设计内容进行全面讲述, 包括 C 语言的基础知识、语法规则、问题分析、算法设计、上机调试及运行。每章包括学习目标、知识点讲解、应用实例、思考题、小结和习题等内容。其中: 学习目标部分明确应掌握的内容及应达到的目标; 知识点部分详细讲解每个知识点, 采用文字和图形结合的方式, 通俗易懂; 例题在选择和设计上紧扣知识点, 并且形式多样, 能够开拓思路, 加深对知识点的掌握; 思考题引导学生动脑思考, 灵活运用所学知识; 小结部分归纳本章重要知识点; 习题包括选择题、填空题和编程题, 可以检验学习的效果。

本书例题都是经过编者精心筛选的, 所有例题程序都已在 Visual C++ 6.0 环境下运行通过, 在其他 C 语言环境下也都可以运行通过。另外, 本书配有电子教案, 并提供例题源程序及课后习题参考答案, 方便读者自学。

本书由冯志红主编, 王春娴副主编。第 1 章、第 6 章和附录由冯志红编写; 第 2 章、第 9 章由王春娴编写; 第 3 章由刘洋编写; 第 4 章由常海燕编写; 第 5 章由马菲编写; 第 7 章、第 10 章由褚益清编写; 第 8 章由李凤荣编写; 第 11 章由刘晶编写; 第 12 章由宁安良编写。全书由冯志红负责统稿和定稿。

本书在编写过程中参考了大量的文献, 在此, 对参考文献的作者表示衷心的感谢! 同时, 对在本书出版中付出努力的清华大学出版社的有关同志表示诚挚的谢意!

由于作者水平有限, 书中难免存在错误和不妥之处, 敬请专家和广大读者批评指正。

编 者

2017 年 1 月

目 录

第1章 C程序设计概述	1
1.1 程序设计概述	1
1.1.1 程序设计语言	1
1.1.2 C语言的发展	3
1.1.3 C语言的特点	4
1.1.4 C语言的应用领域	4
1.2 C程序的结构	5
1.2.1 C程序的基本结构	5
1.2.2 C程序的结构特点	6
1.2.3 C程序的书写规范	7
1.3 算法及其描述	8
1.3.1 算法的概念	8
1.3.2 算法的特性	8
1.3.3 算法的描述	9
1.4 结构化程序设计	11
1.4.1 结构化程序设计方法	11
1.4.2 C语言三大基本结构	12
1.5 C程序的实现过程	13
1.5.1 C程序设计的一般步骤	13
1.5.2 调试运行C程序的过程	13
1.5.3 C语言的编译环境	14
本章小结	18
习题一	19
第2章 数据类型、运算符与表达式	21
2.1 数据类型	21
2.2 标识符、常量与变量	22
2.2.1 关键字与标识符	22
2.2.2 常量与变量的概念	22
2.2.3 整型常量与变量	23
2.2.4 实型常量与变量	25

2.2.5	字符型常量与变量	25
2.2.6	字符串常量	26
2.2.7	变量初始化	27
2.3	运算符与表达式	27
2.3.1	算术运算符与表达式	27
2.3.2	赋值运算符与表达式	29
2.3.3	逗号运算符与表达式	30
2.3.4	测试类型长度运算符	30
2.4	数据类型转换	30
2.4.1	自动类型转换	31
2.4.2	强制类型转换	32
	本章小结	32
	习题二	33
第3章	顺序结构程序设计	35
3.1	C 语言语句	35
3.1.1	声明语句	35
3.1.2	执行语句	36
3.2	格式输入/输出函数	37
3.2.1	格式输出函数 printf()	37
3.2.2	格式输入函数 scanf()	41
3.3	单字符输入/输出函数	45
3.3.1	单字符输出函数 putchar()	45
3.3.2	单字符输入函数 getchar()	45
3.4	顺序结构程序典型例题	47
	本章小结	50
	习题三	50
第4章	选择结构程序设计	55
4.1	关系运算符与关系表达式	55
4.1.1	关系运算符	55
4.1.2	关系表达式	56
4.2	逻辑运算符与逻辑表达式	57
4.2.1	逻辑运算符	57
4.2.2	逻辑表达式	57
4.3	条件运算符与条件表达式	58
4.4	if 语句	60
4.4.1	if 语句的一般形式	60
4.4.2	if 语句的嵌套	65

4.5	switch 语句	67
4.6	选择结构程序典型例题	70
	本章小结	73
	习题四	74
第 5 章	循环结构程序设计	78
5.1	概述	78
5.2	三种循环结构	78
5.2.1	while 循环	78
5.2.2	do-while 循环	80
5.2.3	for 循环	82
5.2.4	几种循环的比较	85
5.3	循环控制语句	85
5.3.1	break 语句	85
5.3.2	continue 语句	87
5.4	循环的嵌套	88
5.5	goto 语句构成的循环	91
5.6	循环结构程序典型例题	92
	本章小结	95
	习题五	95
第 6 章	数组	100
6.1	一维数组	100
6.1.1	一维数组的定义与存储	100
6.1.2	一维数组元素的引用	101
6.1.3	一维数组的初始化	103
6.1.4	一维数组的应用	104
6.2	二维数组	110
6.2.1	二维数组的定义与存储	110
6.2.2	二维数组元素的引用	112
6.2.3	二维数组的初始化	112
6.2.4	二维数组的应用	114
6.3	字符数组与字符串	116
6.3.1	字符串与字符数组的区别	116
6.3.2	字符数组的定义及初始化	117
6.3.3	字符数组的输入与输出	117
6.3.4	字符串的输入与输出	119
6.3.5	字符串处理函数	120
6.3.6	字符数组的应用	122

本章小结	125
习题六	126

第7章 函数	130
--------	-----

7.1 模块化程序设计	130
7.1.1 模块化程序设计概念	130
7.1.2 函数概述	131
7.2 函数的定义与调用	132
7.2.1 函数的定义	132
7.2.2 函数的声明	134
7.2.3 函数的调用	135
7.2.4 函数的参数传递	136
7.2.5 函数应用实例	137
7.3 数组作函数参数	141
7.3.1 数组元素作函数参数	141
7.3.2 数组名作函数参数	141
7.4 函数的嵌套调用与递归调用	146
7.4.1 函数的嵌套调用	146
7.4.2 函数的递归调用	147
7.5 变量的作用域	151
7.5.1 内部变量	151
7.5.2 外部变量	152
7.6 变量的存储方式	155
7.6.1 动态存储与静态存储	155
7.6.2 内部变量的存储方式	155
7.6.3 外部变量的存储方式	157
7.7 内部函数与外部函数	158
7.7.1 内部函数	158
7.7.2 外部函数	159
7.7.3 多个源程序文件的编译与链接	160
本章小结	161
习题七	161

第8章 编译预处理	169
-----------	-----

8.1 宏定义	169
8.1.1 不带参数的宏定义	169
8.1.2 带参数的宏定义	171
8.2 文件包含	172

8.3 条件编译	174
本章小结	175
习题八	175
第9章 指针	178
9.1 地址和指针的概念	178
9.2 指针变量	179
9.2.1 指针变量的定义	179
9.2.2 指针变量的初始化与赋值	180
9.2.3 指针变量的引用	181
9.2.4 指针变量的运算	183
9.3 指针与数组	184
9.3.1 指针与一维数组	184
9.3.2 指针与二维数组	188
9.4 指针与字符串	191
9.4.1 指向字符数组的指针变量	192
9.4.2 指向字符串常量的指针变量	197
9.5 指针与函数	199
9.5.1 指针作函数参数	199
9.5.2 指向函数的指针	202
9.5.3 返回指针值的函数	203
9.6 指针数组和多级指针	205
9.6.1 指针数组的定义	205
9.6.2 指针数组与字符串	205
9.6.3 多级指针	208
9.6.4 指针数组作函数参数	210
本章小结	211
习题九	212
第10章 结构体、共用体和枚举类型	215
10.1 结构体类型	215
10.1.1 结构体类型的定义	215
10.1.2 用 typedef 命名数据类型	217
10.2 结构体变量	218
10.2.1 结构体变量的定义	218
10.2.2 结构体变量的引用	219
10.2.3 结构体变量的初始化	221
10.3 结构体数组	221
10.3.1 结构体数组的定义及初始化	222

10.3.2	结构体数组的应用	223
10.4	结构体类型指针	226
10.4.1	指向结构体变量的指针	226
10.4.2	指向结构体数组的指针	227
10.5	结构体与函数	228
10.5.1	结构体类型数据作函数参数	228
10.5.2	函数的返回值为结构体变量或结构体指针	233
10.6	链表——结构体指针的应用	233
10.6.1	链表的概念	233
10.6.2	动态内存管理函数	235
10.6.3	链表的基本操作	236
10.7	共用体和枚举类型	242
10.7.1	共用体	242
10.7.2	枚举类型	245
	本章小结	246
	习题十	247
第 11 章	文件	251
11.1	文件与文件指针	251
11.1.1	文件的概念	251
11.1.2	文件指针	253
11.2	文件的打开与关闭	254
11.2.1	文件的打开	254
11.2.2	文件的关闭	256
11.3	文件的顺序读/写	257
11.3.1	字符读/写函数	257
11.3.2	字符串读/写函数	259
11.3.3	格式化读/写函数	261
11.3.4	数据块读/写函数	263
11.4	文件的随机读/写	265
11.4.1	文件位置指针的定位	265
11.4.2	文件的随机读/写	267
11.5	文件操作的出错检测	268
11.5.1	ferror 函数	268
11.5.2	clearerr 函数	268
	本章小结	268
	习题十一	269

第 12 章 位运算.....272

12.1 位的基础知识.....272

12.2 位运算概述.....273

12.2.1 位运算符273

12.2.2 位运算符使用方法273

12.2.3 应用举例276

12.3 位段.....278

本章小结.....280

习题十二.....281

附录 A C 语言中的关键字.....283

附录 B 运算符的优先级与结合性.....284

附录 C 常用标准库函数.....286

附录 D C 语言常见语法符.....290

附录 E 标准 ASCII 字符编码表.....292

附录 F 进制及其转换，原码、反码与补码.....294

参考答案.....298

参考文献.....302

第 1 章

C 程序设计概述

学习目标

1. 了解程序设计语言的基本概念及 C 语言的发展、特点和应用领域。
2. 了解 C 程序的基本结构、结构特点及书写规范。
3. 理解算法的概念及特性，掌握算法的常用描述方法。
4. 掌握结构化程序设计的方法及 C 语言三大基本结构。
5. 熟悉 C 程序设计的一般步骤、调试运行 C 程序的过程及 C 语言的编译环境。

计算机之所以能自动、高效地完成各种各样的工作，是因为它能执行事先设计好的相应程序，可以说，计算机的一切操作都是由程序控制的，离开程序，计算机将一事无成。这些程序都是用程序设计语言编写的，根据维基百科的资料，称得上相对“主流”（有人用、有文档）的程序设计语言至少有 600 种，但是，由于 C 语言具有独特的优势，使其得到了广泛的应用。本章主要介绍程序设计概述、C 程序的结构、算法及其描述、结构化程序设计方法以及 C 程序的实现过程。

1.1 程序设计概述

1.1.1 程序设计语言

人与人之间交流使用的语言有很多种，可以是汉语、英语、法语和俄语等。人与计算机交流也通过语言，就是计算机语言。程序设计语言就是一种计算机语言。要了解程序设计语言先要了解指令和程序的概念。指令是指示计算机执行某种操作的命令，程序是一系列按一定顺序排列的指令，程序设计语言是用于编写计算机程序的语言。执行程序的过程就是计算机的工作过程。自 20 世纪 60 年代以来，世界上公布的程序设计语言已有上千种之多，但只有很少一部分得到了广泛的应用。程序设计语言的发展历程可以分为四大阶段，每一个阶段都大大提高了程序设计的效率。

1. 第一代程序设计语言，机器语言——面向机器的语言

机器语言是最底层的计算机语言，又称为二进制语言。用机器语言编写的程序，每一条指令都是“二进制”（0 和 1）形式的指令代码，计算机硬件可以直接识别这些指令代码。

机器语言指令示例：

0000,0000,000000010000 代表 LOAD A,16

0001,0001,000000000001 代表 STORE B,1

因为机器语言程序是直接针对计算机硬件编写的，所以它的执行效率比较高。但对于不同的计算机硬件（主要是 CPU），其机器语言是不相通的，使用一种计算机的机器指令编写的程序，不能在另一种计算机上执行。使用机器语言编写程序，要求编程人员熟记所用计算机的全部指令代码及其含义，因此用机器语言编写程序的难度较大，容易出错，而且程序的直观性较差。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使用机器语言编写程序。

2. 第二代程序设计语言，汇编语言——面向机器的语言

为了便于理解与记忆，人们采用能“帮助记忆”的英文缩写符号（称为指令助记符）来代替机器语言中的指令代码，这种语言称为汇编语言或符号语言。

汇编语言指令示例：

```
MOV AX, BX
```

表示把寄存器 BX 的内容送到 AX 中。

因为汇编语言采用了助记符，所以它比机器语言直观，容易理解和记忆，但是计算机不能直接识别用汇编语言编写的程序，并且汇编语言也依赖于计算机硬件，因此汇编语言程序的可移植性较差。

汇编语言不像其他大多数的程序设计语言一样被广泛用于程序设计。它通常被应用在底层、硬件操作和程序优化要求高的场合。驱动程序、嵌入式操作系统和实时运行程序都需要汇编语言。

3. 第三代程序设计语言，高级语言——面向过程、面向对象的语言

机器语言和汇编语言是低级语言，两种语言都是面向机器的。高级语言基本脱离了机器的硬件系统，具有良好的通用性和可移植性，高级语言的指令接近自然语言和数学公式，编写的程序称之为源程序，计算机不能直接执行源程序，需要翻译成目标程序（二进制程序）才能执行。高级语言包括很多编程语言，如流行的 Java、C、C++、C#、Pascal 等，这些语言的语法、指令格式都不相同。

C 语言指令示例：

```
y=x+10;
```

表示将变量 x 的值和 10 相加，结果存放到变量 y 中。

高级语言分面向过程的语言和面向对象的语言两种。面向过程的语言以过程或函数为基础，在编写程序时需要具体指定每一个过程的细节，这种语言对底层硬件、内存等操作比较方便，C 语言就是面向过程的语言。面向对象的语言由面向过程的语言发展而来，它的操作以对象为基础，但正是这个特性使它对底层的操作不是很方便。可以说面向过程就是什么事都自己做，面向对象就是什么事都指挥对象去做。当程序规模小时用面向过程的语言还可以，当程序规模比较大时用面向对象的语言比较方便。

4. 第四代程序设计语言——非过程化语言

第四代程序设计语言（Fourth-Generation Language, 4GL）是非过程化语言，前三代语言编程时需要指出“怎么做（运行步骤）”，而 4GL 只需要说明“做什么（目标）”，不需描述算法细节。数据库查询和应用程序生成器是 4GL 的两个典型应用。用户可以用数据库查询语言（SQL）对数据库中的信息进行复杂的操作。用户只需将要查找的内容在什么地方、根据什么条件进行查找等信息告诉 SQL，SQL 将自动完成查找过程。应用程序生成器则是

根据用户的需求“自动生成”满足需求的高级语言程序。4GL 是面向应用、为最终用户设计的一类程序设计语言。它具有缩短应用程序开发周期、降低维护成本、最大限度地减少调试过程中出现的问题以及对用户友好等优点。真正的 4GL 应该说还没有出现。目前的 4GL 大多是指基于某种语言环境的具有 4GL 特征的软件工具产品,如 PowerBuilder、FOCUS 等。

1.1.2 C 语言的发展

C 语言的原型是 ALGOL 60 语言(也称为 A 语言)。

1963 年,英国的剑桥大学将 ALGOL 60 语言发展成为 CPL (Combined Programming Language) 语言。

1967 年,英国剑桥大学的 Martin Richards 对 CPL 语言进行了简化,产生了 BCPL 语言。

1970 年,美国贝尔实验室的 Ken Thompson 对 BCPL 语言进行了修改,并为它起了一个有趣的名字 B 语言,还用 B 语言写了第一个 UNIX 操作系统。

1973 年,美国贝尔实验室的 Dennis M. Ritchie 在 B 语言的基础上设计出了一种新的语言,他取了 BCPL 的第二字母作为这种语言的名字,这就是 C 语言。之后不久,他将 UNIX 操作系统的内核和应用程序全部改成用 C 语言编写。

1977 年, Dennis M. Ritchie 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 语言编译程序》。

1978 年, Brian W. Kernighan 和 Dennis M. Ritchie 合著了著名的 *The C Programming Language* 一书,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,人们称之为 K&R C。之后, C 语言被移植到各种机型上,并得到了广泛的支持,使 C 语言在当时的软件开发中几乎一统天下。

1987 年,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,使得不同版本的 C 语言之间出现了一些不一致的地方。为了改变这种情况,美国国家标准局 ANSI (American National Standards Institute) 制定了第一个 C 语言标准,1989 年,这个标准被正式采用,称为 C89,也称 ANSI C。1990 年,该标准被国际标准化组织 ISO (International Standard Organization) 采纳,成为国际标准 (ISO/IEC 9899:1990),称为 C90,也称 ISO C。

1999 年,ISO 制定了新版本的 C 语言标准 (ISO/IEC 9899:1999),称为 C99。

2011 年,ISO 又制定了新版本的 C 语言标准 (ISO/IEC 9899:2011),称为 C11,它是 C 语言的最新标准。

目前流行的 C 语言编译系统大多是以 ANSI C 为基础进行开发的,但不同版本的 C 编译系统所实现的语言功能和语法规则略有差别。

目前, C/C++ 市场占有率较高。C++ 是在 C 语言的基础上发展起来的一种面向对象的编程语言,二者在很多方面是兼容的, C++ 主要在 C 语言的基础上增加了面向对象和泛型的机制,提高了开发效率,因此,更适用于大型软件的编写。C++ 支持面向过程编程、面向对象编程和泛型编程,而 C 语言仅支持面向过程编程。就面向过程编程而言, C++ 和 C 几乎是一样的。由于 C 语言相对简单,各大高校都将 C 语言作为一门重要的课程,学习了 C 语言,再去学习 C++, 就会达到事半功倍的效果。

在 C 语言的发展过程中出现了各种各样的编译系统,常用的有微软公司的 Microsoft Visual C++、GCC 和 Win-TC, Borland 公司的 Turbo C 2.0、Borland C++ 等。这些编译系统

的基本部分是相同的,但也存在着一些差别,所以请读者注意自己所用的 C 编译系统的特点和规则(可参阅相应的手册)。本书使用的上机环境是 Visual C++ 6.0。

1.1.3 C 语言的特点

从第一种高级语言问世到现在已经出现过几百种高级语言了,但是有的很快就被淘汰了,而 C 语言至今一直很流行,因为它具有强大的功能和突出的特点。C 语言的主要特点如下:

(1) C 语言简洁、紧凑,使用灵活、方便。C 语言共有 32 个关键字(见附录 A),9 种控制语句,程序书写自由,主要用小写字母表示。

(2) 运算符丰富,数据处理能力强。C 语言的运算符极其丰富,共有 34 个运算符(见附录 B),包含的范围很广泛,可以实现其他高级语言难以实现的运算。

(3) 数据结构丰富,功能强大。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、共用体类型等,能实现各种复杂数据结构的运算。尤其是指针类型,使程序效率更高。

(4) C 语言是结构化、模块化的语言。C 语言有三大基本结构:顺序、选择和循环,使程序完全结构化。C 语言采用函数作为程序的基本组成单位,便于实现程序的模块化,使程序层次清晰,便于使用、维护及调试。

(5) 语法限制不太严格、程序设计自由度大。例如,对数组下标不做越界检查;整型和字符型通用;一条语句可以分写在多行,也可多条语句写在一行。

(6) 允许直接访问物理地址,可以直接对硬件进行操作。C 语言既具有高级语言的功能,又具有低级语言的许多功能,可以对位、字节和地址进行操作,因此,可以用于编写系统软件,也可对硬件进行编程。

(7) 程序生成的目标代码质量高,执行效率高。

(8) 可移植性好。C 语言适合于多种操作系统,如 DOS、UNIX、Windows 等,也适用于多种机型。

1.1.4 C 语言的应用领域

C 语言是目前使用非常广泛的高级程序设计语言,既可用于编写系统软件又可用于编写应用软件。实际上,C 语言几乎可以应用到程序开发的任何领域。下面是一些常用的应用领域:

(1) 操作系统。Linux 和 UNIX 等操作系统的内核都是用 C 语言编写的,很多运行在电路板上的嵌入式操作系统基本都是用 C 语言结合汇编语言编写的。

(2) 嵌入式应用开发。手机、PAD 等电子产品内部的应用软件很多是用 C 语言开发的。

(3) 复杂运算软件。由于 C 语言本身具有丰富的运算功能,又接近底层语言,所以,可以利用 C 语言编写复杂且高效的运算软件。常用于科学计算的 MATLAB 软件,其中一部分就是用 C 语言编写的。

(4) 游戏开发。利用 C 语言可以开发很多游戏,如贪吃蛇、俄罗斯方块、雷神之锤等。

(5) 图形处理。C 语言具有很强的绘图能力和数据处理能力,可以用于制作动画、绘

制二维图形和三维图形等。

学好任何一门计算机语言都不容易，学习 C 语言也是一样。但是，各种计算机语言之间都是相通的，学好了 C 语言再去学习其他语言就比较容易了。

1.2 C 程序的结构

1.2.1 C 程序的基本结构

下面通过几个简单的示例，了解一下 C 语言程序的基本结构，然后在此基础上了解 C 程序的结构特点。

【例 1-1】 在显示器屏幕上输出一行信息。

```
#include <stdio.h>
void main()                /* main称为主函数，本行是函数头 */
{                          /* 函数体开始 */
    printf("This is the first C program\n"); /* 在屏幕上输出一行信息后换行 */
}                          /* 函数体结束 */
```

程序运行结果：

This is the first C program

说明：程序中/*.....*/部分为注释部分，增加程序的可读性，程序执行时不起作用。

【例 1-2】 从键盘输入两个整数，求它们的和并输出。

```
#include <stdio.h>        /* scanf和printf包含在文件stdio.h中，这两个函数使用频繁，
                           所以该行可省 */
void main()
{
    int a, b, sum;        /* 定义三个整型变量，用于存放输入的两个数及它们的和*/
    scanf("%d, %d", &a, &b); /* 从键盘输入两个数，分别存放到变量a和变量b中 */
    sum=a + b;            /* 对变量a和变量b求和，结果存放到变量sum中 */
    printf("%d\n", sum);  /* 输出和sum，然后换行 */
}
```

程序运行结果：

3,6✓（说明：✓表示从键盘上按Enter键）

9

【例 1-3】 main 函数调用其他函数输出两行*号。

```
void print_star()          /* 用户自己定义的函数，本行是函数头 */
{
```

```
        printf("*****\n"); /* 输出一行*号, 后换行 */
    }
    void main()
    {
        print_star(); /* main函数调用print_star函数 */
        print_star();
    }
```

程序运行结果:

```
*****
*****
```

说明: 函数是实现某个功能的模块代码, 上面三个程序中 `scanf` 和 `printf` 都是系统自带的库函数, C 系统包含很多库函数, 每个函数完成不同的功能, 它们被分类存放在不同的 .h 文件 (头文件) 中, 使用前要在 C 源文件的最前面用 `#include` 将其所在的 .h 文件包含进来, 在调用的时候, 输入正确的参数即可实现函数的功能。常用的标准库函数及其功能见附录 C。库函数不能实现的功能, 用户可以自己定义函数实现, 如例 1-3 中的 `print_star` 函数。

上面三个程序都比较小, 只包含一个源程序, 若一个 C 程序比较大, 可以按功能将其分成多个源程序进行编写。从以上三个程序看出, 函数是程序的基本组成单位, 每一个 C 程序都必须有且只能有一个 `main` 函数 (又称主函数), 其他函数可以没有也可以有多个。函数的顺序没有要求, `main` 函数可以放在其他函数之前, 也可以放在其他函数之后, 但是执行程序时, 都是从 `main` 函数开始, 在 `main` 函数中结束。例 1-1 和例 1-2 只包含一个 `main` 函数, 例 1-3 包含一个 `main` 函数和一个 `print_star` 函数。

每个函数都是由函数头和函数体构成。函数定义的一般形式如下:

```
[函数类型] 函数名 ([形式参数表]) /* 函数头 */
{
    声明部分;
    执行部分;
} /* 函数体 */
```

其中, 函数定义的第一行是函数头部分, 加 “[]” 的部分表示该部分内容可以省略, 函数头下面用 “{ }” 括起来的部分是函数体部分。有关函数的内容将在第 7 章详细介绍。

此外, 一些符号如 “;” “:” “%” “()” 以及空格符和回车符等, 也是 C 程序的重要组成部分, 称之为语法符号简称语法符, 所有语法符并不参与程序运行过程的运算。有些语法符是 C 语言运算符中没有的, 而有些语法符与 C 语言运算符形式一样, 但功能不同。

本书对常见的语法符进行了汇总, 见附录 D, 希望引起读者的注意, 并能正确地使用。

1.2.2 C 程序的结构特点

通过 1.2.1 节的程序示例可以看出, C 语言程序结构具有以下几个主要特点:

- (1) 一个 C 程序由一个或多个源程序文件组成。
- (2) 每个源程序文件可包含一个或多个函数。一个 C 程序必须有且只能有一个 `main`