程序设计基础与C语言

王载新
曾大亮
郭一平

华中理工大学出版社

程序设计基础与C语言

王载新　　曾大亮　　郭一平

华中理工大学出版社

图书在版编目(CIP)数据

程序设计基础与C语言/王载新
武汉:华中理工大学出版社，1999年3月
ISBN 7-5609-1800-x

Ⅰ.程…
Ⅱ.①王… ②曾… ③郭…
Ⅲ.语言-程序设计
Ⅳ.TP312

程序设计基础与C语言
王载新 曾大亮 郭一平
责任编辑 钟小珉
*
华中理工大学出版社出版发行
(武昌喻家山 邮编:430074)
新华书店湖北发行所经销
华中理工大学出版社照排室排版
武汉市科普教育印刷厂印刷
*
开本:787×1092 1/16 印张:19 字数:465 000
1999年3月第1版 1999年3月第1次印刷
印数:1—5 000
ISBN 7-5609-1800-x/TP·293
定价:18.80元
(本书如有印装质量问题,请向出版社发行科调换)

前　言

C 语言是近年来在国内外都得到最迅速推广使用的一种现代通用的程序设计语言。它具有丰富的数据类型、灵活方便的多种运算符、新颖的控制流程和数据结构，以及简洁的表达式。它的处理能力强、运算速度快、目标效率高，具有完善的结构化、模块化程序结构。它既具有高级语言的优点，又具有低级语言的许多特点，因此它既适合于编写系统软件，也适合于编写应用软件。由于C 语言的通用性及可移植性，因此可以在不同的机器上移植C 语言程序。学习和使用C 语言已经是越来越多的人的迫切要求。

在高等院校，非计算机专业选修C 语言课程的学生越来越多，由于总学时的限制，许多专业往往只安排学习一门计算机语言课程。为此，我们编写了这本适合初次学习程序设计的读者学习C 语言的书籍。我们力求用通俗易懂的语言，由浅入深，循序渐进地介绍C 语言及其程序设计。本书不要求读者是否学过一门程序设计语言。由于本书是针对初次学习程序设计的读者编写的，因此我们在介绍C 语言之前简要地介绍了计算机语言方面的基本知识。本书在介绍C 语言如何使用的同时，对程序设计中的基本概念和算法都作了详细介绍。

为了让读者体会程序设计的基本要求与技巧，本书选用了大量不同类型的例题和习题，并且所举例题都是读者易于理解的。我们认为，C 语言虽然适合于编写系统软件，但也适合于编写应用软件，多数非计算机专业的读者可能主要是编写应用软件。要编写系统软件需要有一定的硬件知识。本书介绍的例题不涉及到太多的硬件知识，以便一般的读者也能理解和掌握，有了这个基础，今后需要时再扩展有关的知识是不困难的。

本书介绍的是C 语言的最基本部分。要编写大型的或复杂的程序还应进一步学习C 语言的深入和提高部分(例如，利用C 语言绘图或控制)，请参阅有关书籍。

本书共 10 章，第一、第八、第九、第十章和附录由王载新编写，第二、第三和第四章由曾大亮编写，第五、第六和第七章由郭一平编写。

由于时间仓促及编者水平所限，书中错误和不当之处在所难免，敬请读者批评指正。

编者

1998 年 6 月于华中理工大学

目　录

第一章　C语言概述 …… (1)
1.1　计算机语言 …… (1)
1.2　C语言的发展与特点 …… (1)
1.2.1　C语言的发展 …… (1)
1.2.2　C语言的特点 …… (2)
1.3　简单的C语言程序介绍 …… (3)
1.4　C语言程序的开发过程 …… (5)
1.4.1　在Turbo C集成环境下运行C程序 …… (6)
1.4.2　在Unix操作系统下运行C程序的过程 …… (7)
本章小结 …… (8)
习题一 …… (8)
第二章　基本数据类型和表达式 …… (9)
2.1　保留字和标识符 …… (9)
2.2　基本数据类型 …… (10)
2.2.1　常量和变量的概念 …… (10)
2.2.2　整型变量及其输出 …… (12)
2.2.3　实型变量及其输出 …… (14)
2.2.4　整型常量和实型常量 …… (16)
2.2.5　字符常量和字符串常量 …… (17)
2.2.6　字符变量及其输出 …… (19)
2.2.7　变量的初始化 …… (21)
2.3　运算符和表达式 …… (21)
2.3.1　表达式 …… (21)
2.3.2　算术运算符和算术表达式 …… (22)
2.3.3　算术表达式中数据类型的转换 …… (23)
2.3.4　赋值运算 …… (24)
2.3.5　标准库函数调用 …… (27)
2.3.6　增量运算符与增量表达式 …… (28)
2.3.7　关系运算符和关系表达式 …… (30)
2.3.8　逻辑运算符和逻辑表达式 …… (30)
2.3.9　条件运算符和条件表达式 …… (33)
2.3.10　逗号运算符和逗号表达式 …… (34)
2.3.11　运算符优先级和结合方向 …… (34)
本章小结 …… (36)
习题二 …… (36)
第三章　简单语句和选择语句 …… (40)
3.1　结构化程序设计概述 …… (40)

3.2 scanf 函数和字符输入、输出函数 …… (42)
3.2.1 为何要输入数据 …… (42)
3.2.2 scanf 函数 …… (42)
3.2.3 字符输入函数 …… (45)
3.2.4 字符输出函数 …… (45)
3.3 表达式语句 …… (45)
3.4 复合语句 …… (46)
3.5 if 条件语句 …… (46)
3.5.1 if 结构 …… (47)
3.5.2 if-else 结构 …… (50)
3.5.3 else-if 结构 …… (51)
3.5.4 条件语句的嵌套 …… (54)
3.6 结构化流程图 …… (58)
3.7 switch 语句 …… (61)
3.8 程序设计举例 …… (65)
本章小结 …… (68)
习题三 …… (68)
第四章 循环语句和转移语句 …… (71)
4.1 循环的概念 …… (71)
4.2 for 循环 …… (72)
4.3 while 循环 …… (79)
4.4 do-while 循环 …… (81)
4.5 break 语句 …… (83)
4.6 continue 语句 …… (84)
4.7 多重循环 …… (84)
4.8 go to 语句 …… (89)
本章小结 …… (91)
习题四 …… (91)
第五章 数组 …… (93)
5.1 数组的基本概念 …… (93)
5.2 一维数组的定义和引用 …… (94)
5.2.1 一维数组的定义 …… (94)
5.2.2 一维数组的初始化 …… (95)
5.2.3 一维数组元素的引用 …… (96)
5.3 多维数组的定义和引用 …… (98)
5.3.1 多维数组的定义 …… (98)
5.3.2 多维数组的存储顺序 …… (99)
5.3.3 多维数组的初始化 …… (99)
5.3.4 关于初始化的其他说明 …… (101)
5.3.5 多维数组的引用 …… (101)
5.4 数组的基本操作 …… (103)

5.4.1 一维数组的典型操作 …… (103)
5.4.2 多维数组的操作 …… (107)
5.5 字符型数组 …… (111)
5.5.1 字符数组的定义 …… (112)
5.5.2 字符数组的初始化 …… (112)
5.5.3 字符数组的输入输出 …… (113)
5.5.4 字符串处理函数 …… (115)
5.6 程序设计举例 …… (119)
本章小结 …… (129)
习题五 …… (129)
第六章 函数 …… (133)
6.1 函数定义 …… (133)
6.1.1 函数的分类 …… (133)
6.1.2 函数的基本结构 …… (134)
6.1.3 函数的定义 …… (134)
6.2 函数调用 …… (137)
6.2.1 函数的调用形式 …… (137)
6.2.2 函数的调用方式 …… (138)
6.2.3 对被调用函数的说明 …… (140)
6.3 函数参数及函数间的数据传递 …… (141)
6.3.1 函数参数 …… (141)
6.3.2 函数间的数据传递 …… (142)
6.4 数组与函数参数 …… (147)
6.5 函数的嵌套与递归 …… (151)
6.5.1 函数的嵌套调用 …… (151)
6.5.2 函数的递归调用 …… (154)
6.6 变量的存储属性及作用域 …… (161)
6.6.1 全局变量和局部变量 …… (161)
6.6.2 变量的存储属性及作用域 …… (163)
6.7 编译预处理 …… (167)
6.7.1 宏定义 …… (168)
6.7.2 文件包含 …… (173)
6.7.3 条件编译 …… (174)
6.8 程序设计举例 …… (177)
本章小结 …… (186)
习题六 …… (187)
第七章 指针 …… (189)
7.1 指针的概念 …… (189)
7.1.1 存储单元的地址与内容 …… (189)
7.1.2 指针和指针变量 …… (189)
7.1.3 指针变量的定义 …… (190)
7.1.4 指针运算符和指针变量的引用 …… (191)

7.2 指针运算 …… (195)
7.2.1 指针的算术运算 …… (195)
7.2.2 指针的关系运算 …… (197)
7.2.3 指针变量的赋值运算 …… (198)
7.2.4 指针运算符与自增、自减运算符的混用 …… (198)
7.3 指针与函数参数 …… (200)
7.4 指针与数组 …… (201)
7.5 字符指针和字符串 …… (205)
7.6 函数型指针 …… (208)
7.6.1 函数型指针的定义 …… (208)
7.6.2 函数型指针的赋值 …… (208)
7.6.3 函数型指针的引用 …… (209)
7.7 指针型指针 …… (211)
7.7.1 指针型指针变量的定义 …… (211)
7.7.2 指针型指针的引用 …… (212)
7.8 命令行参数 …… (213)
7.9 程序设计举例 …… (215)
本章小结 …… (218)
习题七 …… (219)
第八章 结构与联合 …… (222)
8.1 结构定义和结构变量的引用 …… (222)
8.1.1 结构定义 …… (222)
8.1.2 结构变量的引用 …… (224)
8.1.3 结构变量的初始化 …… (225)
8.2 结构数组 …… (226)
8.2.1 结构数组的定义 …… (226)
8.2.2 结构数组的初始化 …… (226)
8.3 指向结构的指针 …… (228)
8.4 结构与函数 …… (230)
8.5 引用自身的结构 …… (233)
8.6 字段结构 …… (235)
8.7 联合 …… (236)
8.8 枚举 …… (237)
8.9 类型定义 …… (239)
8.10 程序设计举例 …… (241)
本章小结 …… (244)
习题八 …… (244)
第九章 输入输出 …… (246)
9.1 终端输出函数 …… (246)
9.1.1 字符输出函数 putchar …… (246)
9.1.2 格式输出函数 printf …… (247)

9.2 终端输入函数 …… (252)
9.2.1 字符输入函数 getchar …… (252)
9.2.2 格式输入函数 scanf …… (253)
9.2.3 字符串输入函数 gets …… (255)
9.3 系统命令调用函数 system …… (256)
9.4 程序举例 …… (257)
本章小结 …… (258)
习题九 …… (258)
第十章 文件 …… (261)
10.1 C 文件概述 …… (261)
10.2 文件类型指针 …… (262)
10.3 文件的打开与关闭 …… (263)
10.3.1 文件的打开(fopen 函数) …… (263)
10.3.2 文件的关闭(fclose 函数) …… (264)
10.4 文件的读写 …… (264)
10.4.1 文件的字符读写函数 …… (264)
10.4.2 文件的字符串读写函数 …… (266)
10.4.3 文件的数据块读写函数 …… (267)
10.4.4 文件的格式化输入输出函数 …… (270)
10.4.5 其他读写函数 …… (270)
10.5 文件的定位 …… (272)
10.5.1 置文件位置指针于文件开头位置的函数 rewind …… (272)
10.5.2 改变文件位置指针位置的函数 fseek …… (272)
10.5.3 取得文件当前位置的函数 ftell …… (274)
10.6 文件的错误检测 …… (274)
10.6.1 文件的读写错误检测函数 ferror …… (274)
10.6.2 清除文件错误标志函数 clearerr …… (274)
10.7 程序设计举例 …… (274)
本章小结 …… (278)
习题十 …… (278)
附录 1 常用字符与 ASCII 代码对照表 …… (280)
附录 2 C 语言常用语法提要 …… (281)
附录 3 C 库函数 …… (285)
参考文献 …… (292)

第一章　C语言概述

C语言是一种非常流行和深受程序设计者欢迎的通用程序设计语言。为了适应初次学习程序设计的读者要求，本章先简要介绍C语言的产生和发展，C语言的主要特点；然后介绍简单的C语言程序，C语言程序的开发过程和上机操作。

1.1　计算机语言

语言是人们交换思想的工具，我们日常生活中使用的汉语、英语等称为自然语言。计算机诞生以后，人们要指挥计算机工作便产生了计算机语言。计算机诞生的初期，人们使用的计算机语言仅由0和1代码组成，被称为机器语言。指令是人们指挥计算机进行某种操作的命令。指令的集合称为程序。用机器语言编写的程序难写、难读和难修改，使计算机的推广使用受到了极大的限制，在计算机诞生后的一段时间里只有少数专业人员能使用计算机。随后人们使用便于记忆的符号代替0和1组成的指令，便产生了符号语言（或称汇编语言）。由汇编语言编写的程序要经过汇编程序将其翻译成机器语言程序，计算机才能执行。用机器语言或用汇编语言编写程序（称程序设计）时都离不开具体的计算机指令系统，用它们编制程序技术上复杂，编制程序的效率不高，故被称为低级语言。

随着计算机的发展，50年代中诞生了计算机高级语言，用高级语言编写的程序有易写、易读、易修改的优点，高级语言的出现使计算机的使用得到迅速普及。到目前为止，世界上有数百种高级语言，但常用的不过几十种（如FORTRAN、PASCAL、C、LISP、COBOL等）。用汇编语言或高级语言编写的程序称为源程序，高级语言源程序必须由相应的编译程序将它翻译成相应的汇编语言程序或机器语言程序，经翻译得到的程序称为目标程序。

1.2　C语言的发展与特点

1.2.1　C语言的发展

C语言是国际上广为流行的，具有发展前途的一种程序设计语言，它既适合于编写系统程序又适合于编写应用程序。以前的系统程序主要是用机器语言或汇编语言编写。例如Unix操作系统就是由美国贝尔实验室K. Thompson和D. M. Richie在1969年用机器语言编写成，随后又用汇编语言编写成。由于汇编语言不可移植，并且描述问题的效率不如高级语言，特别是可读性差，所以K. Thompson决定开发一种高级语言来描述Unix系统。1963年英国的剑桥大学推出了CPL语言，但由于规模比较大难以实现。1967年该校的Matin Richards对CPL语言作了简化，推出了BCPL语言。1970年K. Thompson以BCPL语言为基础，又作了进一步简化，设计出很简单而且很接近硬件的B语言（取BCPL的第一个字母），在PDP—11/20上实现B语言并用B写了Unix操作系统和绝大多数实用程序。由于B语言过于简单，数据无类型（或者说只是一种机器字类型），描述能力有限，B编译程序产生的是解释执行的代码，运行速

度慢，故没有流行起来而导致了 D. M. Richie 于 1971 年在 B 语言的基础上开始设计 C 语言（取 BCPL 的第二个字母）。最初的 C 语言只是为描述和实现 Unix 操作系统提供一种工作语言而设计的。1973 年 K. Thompson 和 D. M. Richie 两人合作把 Unix 的 90%以上部分用 C 语言进行了重新编写（即 Unix 第 5 版）。

以后 C 语言多次作了改进，但主要还是在贝尔实验室内部使用，直到 1975 年 Unix 第 6 版面世后，C 语言的突出优点才引起人们的普遍关注。1977 年出现了不依赖于具体机器的 C 语言编译文本——可移植 C 语言编译程序，使 C 移植到其他机器所需要的工作大大简化了，这也推动了 Unix 操作系统迅速地在各种机器上实现。Unix 操作系统在全世界范围内取得了巨大的成功与 C 语言的使用分不开。随着 Unix 的日益广泛应用，C 语言也迅速得到推广，它们在发展过程中相辅相成。1978 年以后，C 语言已先后移植到大、中、小型机及微机上，已独立于 Unix 系统了。现在 C 语言已风靡全世界，成为世界上应用最广泛的几种计算机语言之一。

1983 年，美国国家标准化协会（ANSI）根据 C 语言诞生以来各种版本对 C 语言的发展和扩充，制定了新的标准，称为 ANSI C。1987 年，ANSI 又公布了新标准——87 ANSI C。目前流行的 C 编译程序都是以它为基础的。本书的叙述也基本上以 87ANSI C 为基础。目前广泛流行的各种版本的 C 语言编译系统虽然基本部分是相同的，但其他部分有些不同。在微型机上使用的 Microsoft C、Turbo C、Quick C 等，它们的不同版本又略有差异，因此读者在使用具体 C 编译系统时，还应通过阅读有关手册了解它的具体规定。

1.2.2 C 语言的特点

一种语言之所以能存在和发展，并具有生命力，总是有些不同于（或优于）其他语言的特点。C 语言有以下几个基本特点：

（1）C 语言简洁、紧凑，使用方便、灵活。从 C 语言的名字就体现出它的简洁性。C 语言一共只有 32 个保留字，9 种控制语句，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，相对其他语言源程序短，因此输入程序时工作量少。

（2）C 语言具有高级语言的特点，又具有低级语言的一些功能。它允许直接访问地址，能进行位（bit）运算，可以直接对硬件进行操作。

（3）C 语言是一种结构化程序设计语言，它具有结构化控制语句（如 if-else，while，do while，switch，for 等语句）。C 语言用函数作为程序模块，以实现程序的模块化。因此，C 语言十分有利于实现结构化、模块化程序设计方法。

（4）C 语言运算符丰富。C 语言的运算符包含的范围很广泛，共有 34 种运算符。C 语言把括号、赋值、强制类型转换等都作为运算符处理，从而使 C 语言的运算符类型极其丰富，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）C 语言的数据类型丰富，具有现代化语言的各种数据类型。C 语言的数据类型有：整型、实型、字符型、数组型、指针型、结构型、联合型和枚举型等。能用来实现各种复杂的数据结构。因此，C 语言具有很强的数据处理能力。

（6）C 语言程序中可以使用如 #define、#include 等编译预处理语句，能进行字符串或特定参数的宏定义，以及实现对外部文本文件的读取和合并，同时还具有 #if、#else 等条件编译预处理语句。这些功能的使用有利于提高程序质量和软件开发的工作效率。

（7）C 语言生成的代码质量高。高级语言能否用来描述系统软件，特别是像操作系统、编译程序等，除了决定于语言表达能力以外，还有一个很重要因素就是该语言的代码质量。如果

代码质量低，系统开销会增大，实际上不可行，所以直到现在汇编语言仍然还用来编写部分系统软件。实验表明，C 语言代码效率只比汇编语言代码效率低 10%～20%，而由于用高级语言描述比汇编语言描述问题迅速，可读性好，特别是 C 语言还比较容易移植，其实际效率降低不多，所以 C 语言就成了人们描述系统软件和应用软件比较理想的工具。

(8) C 语言程序的可移植性好。C 语言程序本身不依赖于机器硬件系统，从而便于在硬件结构不同的机种间和各种操作系统中实现程序的移植。

C 语言的优点很多，但也有不足之处应引起注意。C 语言语法限制不太严格，程序设计时自由度大。例如，对数组下标越界不作检查，由程序编写者自己保证程序的正确。C 语言对变量的类型使用比较灵活。例如，整型与字符型和逻辑型数据可以通用。C 语言允许程序编写者有较大的自由度，放宽了对语法的检查。为此，程序员应当仔细检查程序，保证其正确性，而不要过分依赖 C 语言编译程序去查错。因此，C 语言对程序员要求较高。不过，程序员使用 C 语言编写程序时会感到限制少、灵活性大、功能强，可以编写出任何类型的程序。现在，学习和使用 C 语言的人已经越来越多。

1.3 简单的 C 语言程序介绍

用 C 语言编写的程序，称为 C 语言源程序，简称 C 程序。下面介绍几个简单的 C 程序。

【例 1.1】 输出一行信息的 C 程序。

```
main( )
{
printf ("Hello,good morning! \n");
}
```

该程序的作用是输出以下一行信息：

Hello,good morning!

其中，main 表示“主函数”。C 程序是由一个或多个具有相对独立功能的程序模块组合而成，这样的模块称为函数，每个 C 程序必须有一个 main 函数。函数体由大括弧{ }括起来。本例中主函数内只有一个输出函数调用语句，printf 是 C 语言中的输出函数(详见第九章)。双引号内的字符串原样输出。“\n”是换行符，即在输出“Hello,good morning!”后回车换行。语句最后有一分号。

【例 1.2】 计算两数之和的 C 程序。

```
main( )                          /*求两数之和*/
{ int a,b,sum;                   /*变量说明*/
  a=50;b=45;
  sum=a+b;
  printf("sum=%d \n",sum);
}
```

该程序的作用是求两个整数 50 和 45 之和。程序中的第 1、2 行的“/*……*/”表示注释部分。注释是为了提高程序的可读性而增加的，对程序的编译和运行不起作用。注释可以加在程序的任何位置。

第 2 行为说明语句。说明变量 a、b 和 sum 的值为整型数(int)。int 必须小写，它是 integer

(整数)的缩写。说明该句的作用是让编译程序根据变量类型为变量分配存储单元,变量的名字是为存储单元取的名字。在这里a、b、sum各单元存放的数据分别为:50,45,95,如图1.1(a)所示。需要说明的是这里用到的三个变量名所表示的三个内存单元不一定是相邻的,之所以称作变量名,是因为在这种状态下重新赋值后,所存内容会发生变化。例如:

a=123;b=456;sum=a+b;

则a、b、sum的内容便成为如图1.1(b)所示的情况,所以把它们叫做变量。

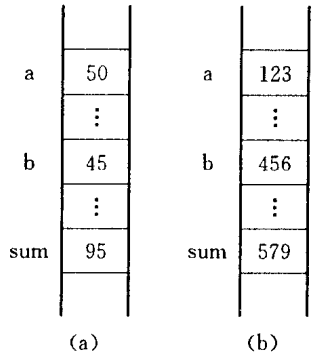

图1.1

第3、4行中出现的"="是赋值运算符,表示把赋值运算符右边的数值或是运算结果(如第4行)赋值给左边的变量。第5行中"%d"是输入、输出的"格式字符串",用来指定输入、输出时的数据类型和格式(详见第九章),"%d"表示十进制整数类型,在执行输出时此位置上代以一个十进制整数值。printf函数中括号内最右端的sum是要输出的变量,现在它的值为95(即50+45之和),因此输出一行信息为:

sum=95

【例1.3】 包含函数调用的C程序。

```
main(  )                          /*主函数*/
{ int a,b;                        /*变量说明*/
  scanf("%d",&a);                 /*输入变量a的值*/
  b=abs(a);                       /*调用abs函数,将得到的值赋给变量b*/
  printf("/%d/=%d\n",a,b);        /*输出a和b的值*/
}
int abs(x)                        /*定义abs函数,函数返回值为整型,x为形
                                    式参数*/
int x;                            /*形参说明*/
{ int y;                          /*abs函数中的变量说明*/
  if (x>0) y=x;
  else y=-x;
  return(y);                      /*将y的值返回调用处*/
}
```

该程序包括两个函数:主函数main和被调用函数abs。abs函数的作用是将x的绝对值赋给变量y。return语句将y的值返回给调用函数main的调用处。main函数中的scanf是"输入函数"的名字(注:scanf和printf都是C语言提供的标准输入、输出函数)。scanf函数括号中变量a前面的"&"的含义是"取地址",这里scanf函数的作用是给以变量a的地址所标志的单元输入数据。关于scanf函数详见第九章。

main函数中第4行为调用abs函数,调用时将实际参数a的值传送给abs函数中的形式参数x。经过执行abs函数得到一个返回值(即abs函数中变量y的值),把这个值赋给变量b。然后输出a和b的值。printf函数中双引号内的"/%d/=%d\n",在输出时,其两个"%d"将先后由a和b的值取而代之,"/ /="原样输出。程序运行结果如下:

−123↙ （输入−123 给 a）

/−123/=123 （输出 a 和 b 的值）

例中用到了函数调用、实参和形参等概念，这里只作了简单的解释。读者如对此不大理解，可以先不予深究，在学到以后有关章节时自然就会理解。在此介绍的这个例子是让读者对 C 程序的组成和形式有一个初步的了解。

通过以上几个例子，可以看到以下几点。

(1) C 程序是由函数构成的。一个 C 程序至少包含 1 个 main 函数，也可以包含一个 main 函数和多个其它函数。被调用的函数可以是系统提供的库函数(例如 printf 和 scanf 函数)，也可以是用户根据需要自己编制的函数(如例 1.3 中的 abs 函数)。

(2) 一个函数由两部分组成：

① 函数的定义部分。包括函数名、函数类型、函数属性、函数形参名、形式参数类型。一个函数名后面必须跟一对圆括号，函数参数可以没有，如 main( )。

② 函数体，即函数说明部分下面的大括号{ }内的部分。如果一个函数内有多对大括号，则最外层的一对{ }为函数体的范围。

函数体一般包括：

a. 变量说明。如例 1.2 中 main 函数中的“int a,b,sum”；

b. 执行部分。由若干个语句组成。

当然，在某些情况下也可以没有变量说明(如例 1.1)。甚至可以既无变量说明，也无执行部分。如：

```
main( )
{ }
```

它是最小的合法的 C 程序，但它什么也未干。

(3) 一个 C 程序总是以 main 函数开始执行的，而不论 main 函数在整个程序中的位置如何，即 main 函数可以放在程序的最前，也可以放在程序的最后，或是放在程序的中间。当然不能放在其他函数中间，因为 C 语言函数不能嵌套定义。

(4) C 程序书写格式自由，语句可以从任一列开始书写，一行内可以写多个语句，一个语句可以分写在多行上。

(5) C 程序中无论是执行语句还是说明语句，每一个语句最后必须有一个分号，即使是程序的最后一个语句也必须有分号，分号是语句结束的标志。

(6) C 语言本身没有输入、输出语句。输入和输出的操作是由库函数 scanf 和 printf 等函数来完成的。

(7) C 程序中可以用/*…*/对任何部分作注释，以增加程序的可读性。注意，注释不能嵌套，如/*…/*…*/…*/是错误的。

1.4 C 语言程序的开发过程

用 C 语言编制程序到完成运行，一般要经过编辑、编译、连接、运行几个阶段，下面对在 Turbo C 环境下和在 Unix 操作系统下运行 C 程序作一简单介绍。

1.4.1 在 Turbo C 集成环境下运行 C 程序

Turbo C 集成开发环境(Turbo C 中的 TC)用起来十分方便,因为它集编辑、编译和调试于一体,无需独立的编辑、编译、连接程序就能建立并运行调试 C 程序。当磁盘上已安装了 Turbo C 时,用其运行 C 程序的步骤如下。

1. 编辑 C 源文件

有两种进入编辑状态的方法。

(1) 设要编辑的源文件名为 file.c,可输入命令:

TC file ↙

以上未输入文件的扩展名,系统自动给文件加上扩展名".c"。如果它是一个新文件,则 Turbo C 屏幕窗口空白等待输入源文件;如果它是一个已存在的文件,则系统将其调入内存并在屏幕窗口上显示其内容,这时用户可以根据需要进行修改。

(2) 输入:

TC ↙

屏幕显示 Turbo C 版本等信息,按回车键后该信息消失,屏幕顶部留下一排"命令"行菜单:

File Edit Run Comple Project Options Debug

用键盘上的"←"和"→"键来移动屏幕上的光标,光标指到哪一个命令字,按回车键就表示执行该命令。开始时,光标指向"FILE",表示对文件进行输入输出。按回车键,"FILE"下面出现一个窗口,如图 1.2 所示。它是一个子菜单,提供多项选择。开始时,光标指向"Load",表示装入文件;按回车键,屏幕又出现一个小窗口(见图 1.3)。其中系统显示 *.c。这时你有两种选择:一是直接输入文件名,按回车键后,系统进入编辑状态,然后可以输入或修改该文件;二是直接按回车键,屏幕将显示当前目录中所有 C 程序文件名,你可以使用光标移动键选择某一文件名,按回车键后该文件被装入即可进行修改。

在图 1.2 所示状态下也可以用"↓"键将光标移到"New"处,按回车键后直接输入 C 程序,系统自动为文件取名 NONAME.C。

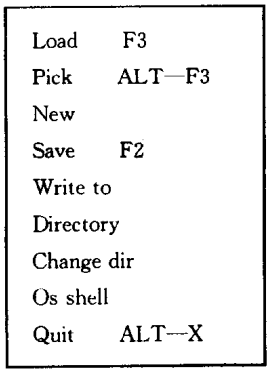

图 1.2

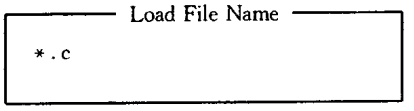

图 1.3

2. 编译、连接、运行

当输入和修改 C 程序后可以同时按"Ctrl"和"F9"键,系统自动执行对 C 程序的编译、连接和运行其生成的可执行目标文件(所生成的目标文件以".OBJ"结尾,可执行目标文件以".EXE"结尾)。在系统发现错误时会自动停下来并在屏幕上显示错误和错误原因。用户可以

根据系统显示的信息修改源程序。再次重复本次操作直到得到正确结果。系统运行完可执行目标文件后立刻回到 TC 屏幕，用户可以同时按下“ALT”和“F5”键回到用户屏上查看运行结果，看完后按任一键系统将回到 TC 屏幕。

3. 保存源文件，退出 Turbo C

程序运行正确后或在编辑文件过程中，可以随时按“F2”键将文件存盘。同时按下“ALT”和“X”键，可以退出 Turbo C 回到 DOS 状态。

Turbo C 的功能很多，详细说明可参阅《Turbo C 用户手册》，下面介绍几种 Turbo C 常用的热键（所谓热键就是能够立即完成某一功能的键）：

F1	显示当前位置的帮助信息
F2	将编辑器里的文件存盘
F3	装入一个文件
F4	程序运行到光标所在行
F5	放大/缩小编辑窗口
F10	切换菜单和编辑窗口
Ctrl—F9	运行程序
Alt—F5	切换 TC 屏幕和用户屏幕
Alt—F6	切换编辑窗口的内容
Alt—F7	定位上一个错误
Alt—F8	定位下一个错误
Alt—E	进入 Edit 菜单
Alt—R	进入 RUN 菜单
Alt—X	退出 TC，返回 DOS

1.4.2 在 Unix 操作系统下运行 C 程序的过程

(1) 用编辑程序（如 Unix 系统下全屏幕编辑程序 vi）将源程序输入计算机，经修改认为无误后存入文件系统。设用户将源文件取名为 f.c。C 程序源文件名一般要以“.c”作为扩展名。

(2) 运行 C 编译程序 cc 对源文件进行编译。可输入命令：

```
cc  f.c↙
```

如果在编译过程中发现源程序中有语法错误，则系统会输出“出错信息”，告诉用户第几行有什么样的错误。用户应重新运行编辑程序，修改后再进行编译。如此下去直到编译通过为止。编译时先生成一个汇编语言程序，即将 C 源程序翻译成为一个汇编语言程序，然后由编译程序调用汇编程序将其翻译成机器语言程序，即目标程序。目标程序的文件名与相应的源程序同名，但将其扩展名自动改为“.o”。上述源文件 f.c 经编译后得到目标程序 f.o。

(3) 连接。将目标程序和库函数或其它目标程序连接成可执行的目标程序。在 Unix 系统下，连接是用 cc 自动完成的。最后得到的可执行目标文件名由系统自动确定为 a.out。

如果不想用系统定的文件名 a.out，也可以在编译时自己指定可执行文件名。例如，想指定为 f.out，可以在编译时输入以下命令：

```
cc    —o    f.out    f.c↙
```

该命令的作用是将源程序 f.c 编译成可执行目标程序，把它存放在 f.out 文件中。此时，a.out 中没有存放本次生成的目标文件。

(4) 运行可执行目标程序，只需输入可执行目标程序文件名即可。例如：

a.out↙　　(系统指定的文件名)

或 f.out↙　　(用户指定的文件名)

运行过程中如果出现运行错误，例如数据溢出等，需再次运行编辑程序，修改源程序重复以上过程。

本章小结

人们利用计算机语言使用和操作计算机。计算机语言分机器语言、汇编语言和高级语言。机器语言和汇编语言又称为低级语言。C 语言属高级语言，但它既具有高级语言的特性，又具有低级语言的功能；既可以用来写应用程序，又可以用来写系统程序。

C 语言简洁、灵活、使用方便。它具有丰富的数据类型，具有结构化控制语句。另外，C 语言程序的可移植性好，所生成代码的质量高。

C 语言程序是由函数构成的，一个 C 程序至少包含一个 main( )函数，也可以包含一个 main( )函数和多个其他函数。这些函数可以放在一个程序文件中，也可以放在多个程序文件中，但是整个程序总是从 main( )主函数开始执行。

C 语言程序的书写格式自由，但所有保留字必须用小写字母表示。程序的语句最后必须有一个分号，分号是语句结束的标志。为了程序层次清楚，便于阅读和理解，最好采用缩进的书写方式。

习 题 一

1.1 简述计算语言的发展过程。

1.2 C 语言的主要用途是什么？它有什么主要特点？

1.3 C 语言程序的主要结构特点和书写格式是什么？

1.4 写出下列程序的输出结果。

```
main( )
{ printf ("Test...");
  printf ("...1");
  printf ("...2");
  printf ("\n");
}
```

1.5 找出下列程序中的错误，然后将修改过的程序输入计算机并进行运行，以验证其是否正确。

```
main( )
{ INT sum;
  /* Compute result
  sum=25+45+50
  /* Display result */
  printf("The answer is %d\n",sum);
}
```