

C语言 程序设计

C Program Design

教程

曾春平 朱小谷 晏海华 编著

红旗出版社



北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

C语言 程序设计

C Program Design

教程

曾春平 朱小谷 晏海华 编著

红旗出版社



北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

TP312C
Z5

图书在版编目 (CIP) 数据

C 语言程序设计教程/曾春平, 朱小谷, 晏海华编著.

北京: 红旗出版社, 2005.3

ISBN 7-5051-1125-6

I. C... II ①曾... ②朱... ③晏... III. C 语言—

程序设计—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字 (2004) 第 128327 号

内 容 简 介

本书是面向 21 世纪高等院校计算机基础系列教材, 主要介绍了利用 C 语言编程的方法和技巧。全书通过对 C 语言概念、数据类型及其运算、基本语句、选择结构程序设计、循环结构程序设计、数组定义和引用、函数、编译预处理、指针、结构体共同体和枚举类型、位运算、文件操作的描述, 介绍 C 语言程序设计的基本方法, 书中列举了大量的程序实例, 帮助读者尽快掌握 C 语言和程序设计的基本方法。

本书可作为各级各类大中专院校相关专业教材, 也可供广大读者学习和巩固 C 语言使用。

需要本书或技术支持的读者, 请与北京中关村 083 信箱 (邮编: 100080) 发行部联系, 电话: 010-82702660, 82702658, 62978181 (总机) 转 103 或 238 传真: 010-82702698 E-mail: tbd@bhp.com.cn。

系 列 名 21 世纪高等院校计算机基础系列教材
书 名 C 语言程序设计教程
编 著 曾春平 朱小谷 晏海华
责 任 编 辑 周凤明 雷 铎
出 版 红旗出版社 北京希望电子出版社
发 行 北京希望电子出版社
地 址 红旗出版社 北京市沙滩北街 2 号 (100727) 电话: (010) 64037138
北京希望电子出版社 北京市海淀区上地三街 9 号金隅嘉华大厦 C 座 610
经 销 各地新华书店 软件连锁店
排 版 希望图书输出中心 孙红
印 刷 北京市媛明印刷厂
版 次 / 印 次 2005 年 3 月第 1 版 2005 年 3 月第 1 次印刷
开 本 / 印 张 787 毫米×1092 毫米 1/16 18.25 印张 419 千字
印 数 1~5000 册
书 号 ISBN 7-5051-1125-6
定 价 24.00 元

21世纪高等院校计算机教材编委会名单

(排名不分先后)

主任:	陈火旺	院士	
副主任:	李仁发	教授	金茂忠 教授 陈 忠 教授 陆卫民 高工
委员:	晏海华	教授	北京航空航天大学
	邵秀丽	副教授	南开大学
	刘振安	教授	中国科技大学
	董玉德	副教授	合肥工业大学
	倪志伟	教授	合肥工业大学
	吕英华	教授	东北师范大学
	杨喜权	副教授	东北师范大学
	朱诗兵	副教授	装备指挥学院
	樊秀梅	副教授	北京理工大学
	徐 安	教授	上海同济大学
	赵 欢	副教授	湖南大学
	胡学钢	教授	合肥工业大学
	林福宗	教授	清华大学
	王家昕	教授	清华大学
	郑 莉	教授	清华大学
	朱淼良	教授	浙江大学
	刁成嘉	副教授	南开大学
	林和平	教授	东北师范大学
	孙铁利	教授	东北师范大学
	温子梅	讲师	广东教育学院
	吕国英	副教授	山西大学
	张广渊	讲师	沈阳大学
	何新华	教授	装甲学院
	邱仲潘	副教授	厦门大学
	曾春平	副教授	第二航空学院
	姬东耀	教授	中科院计算所
	赵宏利	教授	装备指挥学院
	喻 飞	博士	浙江大学
	徐建华	总编	北京希望电子出版社
	郑明红	副总编	北京希望电子出版社
	韩素华	编室主任	北京希望电子出版社
	张 曼	副教授	北京邮电大学
	宋 斌	博士	北京邮电大学
	朱小谷	高级编辑	北京希望电子出版社

总 序

21 世纪挑战与机遇并存, 没有足够的知识储备必将被时代所抛弃。中国 IT 教育产业竞争日趋激烈, 用户需求凸显个性, 行业发展更需要理性。未来五年 IT 行业将以每年 18% 的速度连续增长, 将引发 IT 产业新的发展高潮。实现信息产业大国的目标, 应该依赖教育, 要圆信息产业强国的梦想, 依然要寄托于教育, IT 教育事业任重道远, 其产业也正面临着机遇与挑战。

我国的计算机教学长久以来一直重原理、轻应用。高等院校的计算机教学机制和教材对计算机本身的认识都存在误区。要改革高校计算机教学, 教材改革是重要方面, 用计算机教材的改革促进基础教育的改革势在必行。

一本好书, 是人生前进的阶梯; 一套好教材, 就是教学成功的保证。为缓解计算机技术飞速发展与计算机教材滞后落伍的矛盾, 我们通过调查多所院校的师生, 并多次研讨, 根据读者认识规律, 开创出一种全新的方式, 打破过去介绍原理——理论推导——举例说明的模式, 增加实用操作性, 通过上机实验与课上内容结合来增强可读性, 用通俗易懂的语言和例子说明复杂的概念。

本套教材的特点, 一是“精”, 精选教学内容; 二是“新”, 捕捉最新资讯; 三是“特”, 配备电子课件, 力争达到基础性、先进性、全面性、典型性和可操作性的最大统一。

为保证教材质量, 我们同时聘请了一批学术水平较高的知名专家、教授作为本套教材的主审和编委。全套教材包括必修课教材二十多种, 选修课教材和学习配套用书十余种, 基本上涵盖了目前高等院校(含高等职业技术学院、高等专科学校、成人高等学校)计算机科学与技术专业所必修或选修的内容。各种教材编写时既注意到内容上的连贯性, 又保证了教学上的相对独立性。

本套教材在内容的组织上, 大胆汲取当今计算机领域最新技术, 摒弃了传统教材中陈旧过时的内容。这些变化在各本教材中都得到了不同程度的体现。本套教材编写时既参照了教育部有关计算机科学与技术专业的教学要求, 又参考了“程序员考试大纲”和“全国计算机水平等级考试大纲”的内容, 因此既适合作为高等学校计算机科学与技术专业教材, 也可作为计算机等级考试学习用书。

考虑到各校教学特点和计算机设备条件, 我们本着“学以致用”的理念, 在本套教材编写中自始至终贯彻“由浅入深, 理论联系实际”的原则, 以阐明要义为主, 辅之以必要的例题、习题和上机实习, 能够使学生尽快领悟和掌握。

在本套教材编写过程中, 作者们付出了艰辛的劳动, 教材编委会的各位专家、教授进行了认真的审定和悉心的指导。书中参考、借鉴了国内外同类教材和专著, 在此一并表示感谢。

我们希望更多的优秀教师参与到教材建设中来, 真诚希望广大教师、学生与读者朋友在使用本丛书过程中提出宝贵意见和建议。

若有投稿或建议, 请发至本丛书出版者电子邮件: textbook@bhp.com.cn

21 世纪高等院校计算机教材编委会

目 录

第 1 章 C 语言概述1	3.2 数据的输入和输出及输入输出 34
1.1 C 语言发展简介1	函数的调用..... 34
1.2 C 语言的特点2	3.2.1 scanf()函数(格式化输入函数).. 35
1.2.1 C 语言是中级语言.....2	3.2.2 printf()函数..... 37
1.2.2 C 语言是结构化语言.....3	3.2.3 getchar()函数与 putchar()函数.. 40
1.2.3 C 语言是程序员的语言.....3	3.3 顺序结构程序设计 41
1.3 C 语言的结构4	3.4 习题 42
1.4 C 语言的开发步骤和关键字.....8	第 4 章 选择结构程序设计 47
1.5 习题.....9	4.1 用 if 语句实现选择结构 47
第 2 章 数据类型及其运算10	4.1.1 if 语句的两种基本形式 47
2.1 C 的数据类型及其定义方法.....10	4.1.2 if...else if 语句 49
2.1.1 C 的数据类型.....10	4.2 用 switch 语句实现多分支选择结构 51
2.1.2 常量和变量11	4.3 选择结构的嵌套 53
2.1.3 整型数据14	4.3.1 if 语句嵌套 53
2.1.4 实型数据15	4.3.2 switch 语句嵌套 54
2.1.5 字符型数据16	4.4 习题 54
2.2 运算的种类、优先级和结合性19	第 5 章 循环结构程序设计60
2.2.1 算术运算19	5.1 while 和 do while 循环结构..... 60
2.2.2 关系运算和逻辑运算21	5.1.1 while 循环语句..... 60
2.2.3 位运算23	5.1.2 do-while 循环语句 61
2.2.4 赋值运算23	5.2 for 循环语句..... 63
2.2.5 其他运算24	5.2.1 for 循环语句的一般形式 63
2.2.6 运算符优先级与表达式求值	5.2.2 for 循环变量..... 64
次序.....26	5.2.3 无限循环 65
2.3 数据类型的转换27	5.2.4 没有循环体的 for 循环语句 66
2.3.1 自动类型转换27	5.3 循环嵌套 66
2.3.2 强制类型转换28	5.4 continue 和 break 语句..... 67
2.3.3 赋值表达式中的类型转换.....28	5.4.1 continue 语句..... 67
2.4 习题.....29	5.4.2 break 语句..... 68
第 3 章 基本语句32	5.5 goto 语句和语句标号的使用 69
3.1 表达式语句、空语句及复合语句33	5.6 例题分析 70
3.1.1 表达式语句33	5.7 习题 72
3.1.2 空语句33	第 6 章 数组的定义和引用 76
3.1.3 复合语句33	6.1 一维数组定义、初始化和引用..... 76
3.1.4 带标号的语句34	6.1.1 一维数组的定义..... 76

6.1.2 一维数组的初始化	77	8.1.1 #define	118
6.1.3 一维数组的引用	78	8.1.2 定义带参的宏	119
6.2 二维数组定义、初始化和引用	80	8.2 文件包含	120
6.2.1 二维数组的定义	80	8.3 条件编译命令	120
6.2.2 二维数组的初始化	82	8.4 习题	123
6.2.3 二维数组的引用	83	第 9 章 指针	127
6.2.4 多维数组	84	9.1 地址、指针和指针变量的概念	127
6.3 字符数组	84	9.2 指针与变量	129
6.3.1 字符数组的定义	84	9.2.1 指针变量的定义和类型	129
6.3.2 字符数组的初始化	85	9.2.2 指针变量的引用和运算	129
6.3.3 字符数组的引用	85	9.2.3 指针和指针变量作为函数的 参数	131
6.3.4 字符数组的输入输出	86	9.3 指针与数组	132
6.3.5 字符串数组	88	9.3.1 指向一维数组及其元素的 指针和指针变量	132
6.4 习题	89	9.3.2 一维数组名作函数参数	134
第 7 章 函数	94	9.3.3 二维数组的指针和指向二维 数组的指针变量	136
7.1 标准库函数的正确调用	95	9.3.4 二维数组的指针或指针变量 作为函数的参数	139
7.2 函数的定义	96	9.3.5 字符数组的指针和指向字符 数组的指针变量	140
7.3 函数的作用域规则	97	9.4 指针与函数	142
7.4 函数的类型和返回值	98	9.4.1 函数的指针和指向函数的 指针变量	142
7.5 函数的参数	99	9.4.2 返回指针值的函数	145
7.6 函数的调用	100	9.5 指针数组与指向指针的指针变量	145
7.6.1 函数的正确调用	100	9.5.1 指针数组	145
7.6.2 函数原型——对被调用函数 的声明	101	9.5.2 指向指针数据的指针变量	148
7.6.3 函数的嵌套调用	102	9.5.3 指针数组作 main 函数中 的参数	149
7.6.4 函数的递归调用	104	9.6 习题	150
7.7 内部函数与外部函数	106	第 10 章 结构体、共用体、枚举 类型和自定义类型	155
7.7.1 内部函数（静态函数）	106	10.1 结构体类型及其变量的定义和 引用	155
7.7.2 外部函数	107	10.1.1 结构体类型的定义	155
7.8 局部变量、全局变量及变量的 存储类别	107		
7.8.1 局部变量	107		
7.8.2 全局变量	109		
7.8.3 变量的存储类别	110		
7.9 习题	114		
第 8 章 编译预处理	118		
8.1 宏定义	118		

10.1.2 结构体变量的定义	156	10.7.2 枚举变量的输入输出	180
10.1.3 结构体变量的初始化	158	10.8 自定义数据类型名	182
10.1.4 结构体变量的引用	159	10.8.1 自定义数据类型名的定义	182
10.1.5 结构体变量的输入输出	160	10.8.2 常见自定义类型名举例	183
10.2 结构体数组	161	10.9 习题	184
10.2.1 结构体数组的定义	161	第 11 章 位运算	188
10.2.2 结构体数组的初始化	162	11.1 位运算的含义	188
10.2.3 结构体数组的引用	163	11.2 简单位运算	189
10.3 指向结构体类型数据的指针	166	11.2.1 按位与运算	189
10.3.1 指向结构体变量的指针变 量的定义和引用	166	11.2.2 按位或运算	190
10.3.2 通过（指针或）指针变量 访问结构体数组	167	11.2.3 按位异或运算	190
10.4 结构体与函数	169	11.2.4 取反运算	190
10.4.1 结构体变量作为函数参数 ——单向值传递	169	11.2.5 左移运算与右移运算	190
10.4.2 结构体变量的指针（或指针 变量）作为函数参数——传 地址	169	11.3 习题	192
10.4.3 返回结构体类型值的函数	170	第 12 章 文件操作	193
10.5 简单（单向）链表的建立、 输出、删除和插入	171	12.1 C 文件概述	193
10.5.1 什么是单向链表	171	12.2 文件类型指针	194
10.5.2 建立动态单向链表	171	12.3 文件的打开与关闭	195
10.5.3 删除单向链表的结点	173	12.3.1 文件的打开（fopen 函数）	195
10.5.4 向单向链表中插入结点	174	12.3.2 文件的关闭（fclose 函数）	196
10.5.5 综合操作单向链表	175	12.4 文件的读写	196
10.6 共用体	176	12.4.1 读写字符（fgetc 和 fputc 函数）	197
10.6.1 共用体的概念	176	12.4.2 使用 feof() 函数	199
10.6.2 共用体的定义	176	12.4.3 读写字符串（fgets 和 fputs 函数）	199
10.6.3 共用体变量的引用	177	12.4.4 格式化读写（fscanf 和 fprintf 函数）	201
10.7 枚举类型	179	12.4.5 块读写（fread 和 fwrite 函数）	202
10.7.1 枚举类型和枚举变量的定义 和引用	179	12.5 文件的定位	205
		12.6 习题	207
		习题和参考解答	211
		参考文献	284

第1章 C语言概述

目标

- 了解 C 语言的起源
- 了解 C 语言程序的特点
- 了解 C 语言程序的开发步骤
- 掌握 C 语言程序的结构
- 掌握 main 函数在 C 语言程序中的重要作用
- 了解标准库函数与自定义函数的区别
- 了解头文件对于在 C 程序中使用标准库函数的重要性
- 掌握在 C 程序中进行注释的方法
- 了解 C 程序中函数的基本形式
- 了解 C 程序中的关键字

1.1 C 语言发展简介

关于 C 语言的发展，可以追溯到上个世纪六十年代初。1963 年英国的剑桥大学推出了 CPL (combined programming language) 计算机语言。随后，在 1967 年英国剑桥大学的 Martin Richards 开发出 BCPL (basic combined programming language) 语言。到了 1970 年，美国贝尔实验室的 Ken Thompson 以 BCPL 语言为基础设计出了很简单的 B 语言(取 BCPL 的第一个字母)，还用 B 语言编写了第一个 UNIX 操作系统，并在 PDP-7 计算机上实现。但 B 语言过于简单，功能有限。于是，在 1972 年至 1973 年期间，美国贝尔实验室的 Dennis Ritchie 在 B 语言的基础上设计出了 C 语言(取 BCPL 的第二个字母)，并首先在运行 UNIX 操作系统的 DEC PDP-11 计算机上使用。

在此后的一些年里，C 语言事实上的标准是 UNIX 操作系统第 5 版中提供的版本，并首先在《The C Programming Language》中加以介绍，该书是 C 语言发展史上影响深远的名著，于 1978 年由 Brian Kernighan 和 Dennis Ritchie (简称 K&R) 发表。1983 年，美国国家标准协会 (ANSI) 开始制定 C 语言的标准，称为 ANSI C。制定该标准的过程持续了 6 年，ANSI C 标准终于在 1989 年 12 月正式投入使用，并于 1990 年被国际标准化组织 ISO (International Standard Organization) 接受为 ISO C 的标准。现在，所有主流 C 编译器都符合 ANSI C 标准，但不同版本的 C 略有差异。因此，在使用 C 语言之前，应该了解所用计算机系统配置的 C 编译系统的特点和规定。

1.2 C 语言的特点

1.2.1 C 语言是中级语言

C 语言通常称为中级计算机语言。中级语言并没有贬义，不是说它的功能差、难以使用或者比 BASIC、Pascal 那样的高级语言原始，也不是说它与汇编语言相似，会给使用者带来类似的麻烦。C 语言之所以被称为中级语言，是因为它把高级语言易于理解的成分同汇编语言灵活高效的特点结合起来了。表 1-1 表明了 C 语言在计算机语言中所处的地位。

表1-1 C语言在计算机语言中的地位

高级	Ada Modula-2 Pascal COBOL FORTRAN BASIC
中级	C FORTH Macro-assembler
低级	Assembler

作为中级语言，C 允许对位、字节和地址这些计算机功能中的基本成分进行操作。C 语言程序非常容易移植。可移植性表现为在某种计算机上写的软件可以用到另一种机器上去。举例来说，如果为苹果机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。所有的高级语言都支持数据类型的概念。一个数据类型定义了一个变量的取值范围和可在其上操作的一组运算。常见的数据类型是整型、字符型和实型。虽然 C 语言有 5 种基本数据类型，但与 Pascal 或 Ada 相比，它却不是强类型语言。C 程序允许几乎所有的类型转换。例如，字符型和整型数据能够自由地混合在大多数表达式中进行运算。这在强类型高级语言中是不允许的。

C 语言的另一个重要特点是它仅有 32 个关键字，这些关键字就是构成 C 语言的命令。和 IBM PC 的 BASIC 相比，后者包含的关键字多达 159 个。

1.2.2 C 语言是结构化语言

不严格地讲，C 语言是块结构（block-structured）语言，但是 C 语言也常被称为结构化语言。这是因为它在结构上类似于 ALGOL、Pascal 和 Modula-2（从技术上讲，块结构语言允许在过程和函数中定义过程或函数。用这种方法，全局和局部的概念可以通过“作用域”规则加以扩展，“作用域”管理变量和过程的“可见性”。因为 C 语言不允许在函数中定义函数，所以，严格地讲，不能称之为通常意义上的块结构语言）。

结构化语言的显著特征是代码和数据的分离。这种语言能够把执行某个特殊任务的指令和数据从程序的其余部分分离出去、隐藏起来。获得隔离的一个方法是调用使用局部（临时）变量的子程序。通过使用局部变量，我们能够写出对程序其它部分没有副作用的子程序。这使得编写共享代码段的程序变得十分简单。如果开发了一些分离很好的函数，在引用时我们仅需要知道函数做什么，不必知道它如何做。切记，过度使用全局变量（可以被

全部程序访问的变量)会由于意外的副作用而在程序中引入错误。

结构化语言比非结构化语言更易于程序设计,用结构化语言编写的程序结构清晰,易于理解,因而易于维护。这是人们已经普遍接受的观点。C语言的主要结构成分是函数,即C的独立子程序。

在C语言中,函数是一种构件(程序块),是完成程序功能的基本构件。函数允许一个程序的诸任务被分别定义和编码,使程序模块化。可以确信,一个好的函数不仅能正确工作,而且不会对程序的其它部分产生副作用。

1.2.3 C语言是程序员的语言

初学者常常会认为“所有的计算机语言都是由程序员使用的”,实际并非如此。我们来考察一下典型的非程序员语言 COBOL 和 BASIC。COBOL 的设计使程序员难以改变所编写代码的可靠性,甚至不能提高代码的编写速度。然而,COBOL 设计者的本意却是打算使非程序员能读懂程序(这是不大可能的事)。注意,这并不是说 COBOL 语言不好,而是想指出,它没有被设计成为程序员的理想语言。BASIC 语言的主要目的是允许非专业程序员在计算机上编程解决比较简单的问题。与它们形成鲜明对照的是 C 语言,由于程序生成、修改和现场测试自始至终都能满足程序员的理想要求,因而它实现了程序员的期望:很少限制、很少强求、块结构、独立的函数以及紧凑的关键字集合。用 C 语言编程,程序员可以获得高效机器代码,其运行效率几乎接近汇编语言代码。

C 语言被程序员广泛使用的另一个原因是,可以用它代替汇编语言。汇编语言使用的汇编指令,是能够在计算机上直接执行的二进制机器码的符号表示。汇编语言的每个操作都对应为计算机执行的单一指令。虽然汇编语言给予程序员达到最大灵活性和最高效率的潜力,但开发和调试汇编语言程序的困难是难以忍受的。非结构性使得汇编语言程序难于阅读、改进和维护。也许更重要的是,汇编语言程序不能在使用不同 CPU 的机器间移植。

最初,C语言被用于系统程序设计。“系统程序”是一大类程序的一部分,这一大类构成了计算机操作系统及实用程序。通常被称为系统程序的有:

- 操作系统。
- 翻译程序。
- 编辑程序。
- 汇编程序。
- 编译程序。
- 数据库管理程序。

随着 C 语言的普及,加之其可移植性和高效率,许多程序员用它设计各类程序。几乎所有的计算机上都有 C 语言编译程序,这使我们可以很少改动甚至不加改动地将为一种机器写的 C 语言源程序在另一种机器上编译执行。可移植性节省了时间和财力。

C 语言不仅在速度和结构上有它的优势,而且每个 C 语言系统都提供了专门的函数库,程序员可以根据不同需要对其进行剪裁,以适应各种程序的设计。由于它允许(更准确地说鼓励)分别编译,所以 C 语言可使程序员方便地管理大型项目,最大限度地减少重复劳动。

1.3 C 语言的结构

可以从学习几个简单的例子程序入手来说明 C 语言的结构。例如，在计算机屏幕上打印如下一行信息：

I want to study C language.

对于初学者，可以利用一个编译器（例如，Turbo C）在老师的指导下，先建立程序文本，然后成功地对它编译、连接和运行，最后再看看所产生的输出。这样，可以对整个编写和执行一个 C 程序的过程有一个形象和直观的认识。只要把这个过程中的操作细节掌握了，其它内容就比较容易了。

在 C 语言中，用如下程序打印 I want to study C language.:

[例 1-1]

```
#include <stdio.h>
```

```
main ()
```

```
{
```

```
    printf ("I want to study C language.\n");
```

```
}
```

对于不同的编译器，运行这个程序的步骤会有差别。以 Turbo C 编译器为例，在 Turbo C 编译器中，必须先建立（编辑）这个程序，再以“.c”作为扩展名保存编辑好的文件，如 firstc.c，然后利用 Compile 菜单项中的 build all 命令，可以连续完成编译和连接过程。如果程序中没有错误，将产生一个 firstc.exe 文件。最后以 Run 命令运行这个程序，或者在 DOS 操作系统下输入命令

firstc

则系统将在屏幕上打印

I want to study C language.

这个程序虽简单，但却包含了 C 程序的基本成分。下面是该程序有注释的情况：

```
#include <stdio.h>    /*包含有关标准库的信息，如printf()的定义信息*/
```

```
main ()              /*定义名为main的函数，它不接受外部参数值*/
```

```
{                    /* main函数的语句在花括号中*/
```

```
    printf ("I want to study C language.\n");
```

```
    /*main函数调用标准库函数printf可以打印字符序列，\n表示换行*/
```

```
}
```

程序中/*.....*/表示对程序的说明（称为注释），不参与程序的运行。注释文字可以是任意字符，如汉字、拼音、英文等。

不论大小如何，C 程序都是由函数和（或）变量组成。函数中包含若干语句，语句用于指定所要进行的计算操作。变量用于在计算过程中保存有关的值。本例中，从书写形式上看，程序主要由一个 main 函数组成。一般来说，函数可以任意命名，但 main 是一个特殊的函数名，每个 C 程序都从名为 main 的函数开始执行（故 main 函数被称为 C 语言的入口

函数), 其他函数都是被 main 函数直接或间接调用的函数。例如, 本例中 printf 函数是被 main 函数直接调用的函数。

main 函数的内容要靠用户自己编写, main 函数调用的函数既可以是用户根据需要自己编写的 (自定义函数), 也可以是 C 语言系统标准函数库提供的。例如, 本例中 printf 函数是标准库函数。如果使用标准库函数, 必须把包含标准函数定义信息的头文件 (以.h 为扩展名的文件) 包含到程序中。譬如, 上述程序的第一个行

```
#include <stdio.h>
```

用于通知编译器在本程序中包含标准输入输出的有关信息。#include 是一个编译预处理命令, 用于把其后的头文件 (如 stdio.h) 的内容包括到程序中。本例中, 关键是要把 printf 函数的定义包含到程序 firstc.c 中。printf 是一个用于打印输出数据的标准库函数, 它的函数定义在头文件 stdio.h 中。为了使用已有定义信息的函数 (也可以是用户编写的函数, 其定义存放在以.h 为扩展名的头文件), 可以用#include 命令把存放这些函数定义的头文件包含到程序中, 这样就可以在程序中调用这些函数了。所以, 许多 C 源程序的开始处都有这样的编译预处理命令。

在函数之间进行数据通信的主要方法是让主调函数向被调用函数 (若函数 A 调用函数 B, 则称函数 A 为主调函数, 称函数 B 为被调用函数) 提供一系列的参数值, 被调用函数把运算结果以返回值的形式传递给主调函数。函数名后面的一对圆括号用于把这些参数括起来。在本例中, 所定义的 main 函数不要求任何参数, 所以用空参数表 () 表示。

函数中的语句用花括号括起来。本例中 main 函数只包含一条语句:

```
printf ("I want to study C language.\n");
```

当调用一个函数时, 要先给出这个函数名, 再紧跟着用圆括号括起来的参数表。上面的语句就是用 I want to study C language.\n 作参数来调用函数 printf。用双引号括起来的字符序列叫做字符串或字符串常量, 如 I want to study C language.\n 就是一个字符串, \n 是转义字符, 表示换行, 即把光标移到下一行最左边位置处。

再看两个稍微复杂一些的例子。

[例 1-2]

```
#include <stdio.h>
```

```
main ()          /*主函数*/
{
    void proc (); /*声明函数proc*/
    int a;        /*说明a为整型变量*/
    scanf ("%d",a); /*调用标准库函数scanf, 从键盘输入a的值*/
    proc ();      /*调用函数proc, 无返回*/
    a = fun ();   /*调用函数fun, 结果返回给a */
    printf ("This is a sample of c program.\n");
}
void proc ()      /*定义函数proc, void指定该函数不返回结果*/
{
```

```

printf ("Hello.\n") ;
}
int fun ()          /*定义函数fun, int指定该函数返回一个整数*/
{
    return (8) ;    /*返回整数8*/
}

```

本程序的执行过程是：

- 程序从 main () 处开始。
- 变量 a 代表一个整数。
- scanf 是头文件 stdio.h 中定义的标准输入函数。%d 表示 a 以整型格式输入。
- 执行程序 (函数) proc ()；屏幕上显示 Hello，转义字符 \n 引起换行。
- 执行程序 (函数) fun ()；并将结果赋予 a，此时，a 的值为 8。
- 屏幕上显示 “This is a sample of c program.”。

程序执行的结果是在屏幕上显示两行信息：

Hello.

This is a sample of c program.

[例 1-3]

/*输入长方体的长、宽、高，计算长方体体积*/

```

main ()
{
    int x,y,a;          /*说明x、y、a为整型变量*/
    scanf ("%d,%d", &x, &y) ;    /*调用标准函数，从键盘输入x,y,z的值*/
    a = area (x,y) ;     /*调用area函数，计算面积*/
    printf ("a = %d\n",a) ;
}

int area (int x1,int y1) /*定义area函数，对形参x1、y1作类型定义*/
{
    int p;              /*定义函数内部使用的变量p*/
    p = x1*y1;          /*计算面积的值*/
    return (p) ;        /*将p值返回调用处*/
}

```

本程序的功能是对从键盘输入的长方形的长和宽两个整型量，并求其面积的值。程序运行的情况如下：

5,8<回车>

a = 40

在本例中，main 函数在调用 area 函数时，将实际参数 x、y 的值分别传送给 area 函数中的形式参数 x1、y1。经过执行 area 函数得到一个结果 (即 area 函数中变量 p 的值)，并把这个值赋给变量 a。

从上面的程序例子可以看出 C 程序的基本结构。

C 程序为函数模块结构，所有的 C 程序都是由一个或多个函数构成的，也就是说，函数是构成 C 源程序的基本单位。一个 C 源程序必须且只能包含一个 main 函数（主函数），main 函数既是 C 程序开始执行的入口，也是 C 程序结束执行的出口。具体地说就是：程序从主函数开始执行，当执行到调用函数的语句时，程序将控制转移到被调用函数中执行，执行结束后，再返回主调函数中继续运行，直至程序执行结束。C 程序的函数包括编译系统提供的标准函数（如 printf、scanf 等）和由用户自己定义的函数（如 proc、fun、area 等）。主函数是 C 语言的一个特殊函数，主函数名 main 既不能用作变量名，也不能被修改。

函数在 C 语言中非常重要。函数的基本形式是：

函数类型 函数名（形式参数）

形式参数说明；

{

数据说明部分；

语句部分；

}

其中，函数首部包括函数类型说明、函数名和圆括号中的形式参数列表（如 int area (int x1,int y1)），如果函数调用无参数传递，圆括号中的形式参数为空（如 main（）函数）。函数体包括函数体内使用的数据说明和执行函数功能的语句，花括号{和}表示函数体的开始和结束。

数据说明部分用于说明函数中要使用的函数和变量的类型和名字。例如，在[例 1-2]中

```
void proc () ;    /*声明函数proc*/
```

```
int a;           /*说明a为整型变量*/
```

函数中的数据说明和语句的最后必须有一个分号。例如，

```
int p;
```

```
p = x1*y1;
```

从[例 1-1]至[例 1-3]中可以看到，程序数据的输入输出是通过标准库函数 scanf 和 printf 完成的。为了便于程序移植，C 语言利用标准库函数实现数据的输入输出。所以，C 语言本身没有输入输出语句。

与 FORTRAN 等其他高级语言不同，C 程序的书写格式比较自由。C 程序中没有行号。如果一条语句较长，为了阅读方便，可以写在多行上。如果语句较短，为了节省篇幅，也可以在一行写几个语句。

1.4 C 语言的开发步骤和关键字

开发一个 C 程序（在 Turbo C 环境下）大概包括以下四步。

（1）程序设计：程序设计亦称程序编辑。程序员用 Turbo C 编辑器将编写好的 C 程序输入计算机，并以文本文件的形式保存在计算机的磁盘上。编辑的结果是建立 C 源程序文件。C 程序习惯上使用小写英文字母表示变量，常量和其他用途的符号可用大写字母。C 语言对大、小写字母是有区别的。关键字必须小写。

(2) 程序编译：编译是指将编辑好的源文件翻译成二进制目标代码的过程。编译过程是使用 C 语言提供的编译程序（编译器）完成的。编译时，编译器首先要对源程序中的每一个语句检查语法错误，当发现错误时，就在屏幕上显示错误的位置和错误类型的信息。此时，要再次调用编辑器进行查错修改。然后，再进行编译，直至排除所有语法和语义错误。正确的源程序文件经过编译后在磁盘上生成目标文件。

(3) 连接程序：编译后产生的目标文件是可重定位的程序模块，不能直接运行。连接就是把目标文件和其他分别进行编译生成的目标程序模块（如果有的话）及系统提供的标准库函数链接在一起，生成可以运行的可执行文件的过程。连接过程使用 C 语言提供的连接程序（链接器）完成，生成的可执行文件存在磁盘中。

(4) 程序运行生成可执行文件后，就可以在操作系统控制下运行。若执行程序后达到了预期目的，则 C 程序的开发工作到此完成。否则，要进一步检查修改源程序，重复编辑—编译—链接—运行的过程，直到取得预期结果为止。

在编写 C 语言程序时，应该注意有些词汇是 C 语言中的关键字，这些关键字在 C 语言中具有特殊的含义，不能被当作函数名或变量名等标识符使用。这些关键字是：

auto	break	case	char	const
continue	default	do	double	else
enum	extern	float	for	goto
if	int	long	register	return
short	signed	sizeof	static	struct
switch	typedef	union	unsigned	void
volatile	while			

C 语言的关键字都用小写字母。C 语言中区分大写与小写，else 是关键字，而 ELSE 则不是。

小结

开发 C 语言程序分 4 个步骤：第一步是用编辑器编辑 C 源程序文件；第二步是把 C 源程序文件编译成二进制代码的目标文件；第三步是把目标文件（可以是多个）和标准库函数进行连接并生成可执行文件；第四步是运行可执行文件。

C 语言程序是函数式模块结构，即函数是构成 C 源程序的基本单位。一个 C 源程序必须且只能包含一个 main 函数（主函数），main 函数既是 C 程序开始执行的入口，也是 C 程序结束执行的出口。

自定义函数是用户自己编写的函数，标准库函数是 C 语言系统标准函数库提供的。如果在程序中使用标准库函数，必须把包含标准函数定义信息的头文件（以.h 为扩展名的文件）包含到程序中。

在函数之间进行数据通信主要是通过函数的参数值和返回值来实现的。函数由函数首部和函数体组成。函数首部包括函数类型说明、函数名和圆括号中的形式参数列表。函数体用花括号{和}括起来，花括号中包括数据说明和语句，数据说明集中在一起放在前面，语句集中在一起放在后面。

这些关键字在 C 语言中具有特殊的含义，不能被当作函数名或变量名等标识符使用。C

语言的关键字都用小写字母。

1.5 习 题

1. 一个 C 程序的执行顺序是：从本程序的 main 函数开始，到本程序的 main 函数结束。
2. 语句 是组成 C 程序的基本单位。
3. 下述正确的叙述是（多选）BCDE。
 - (1) ☒ 一个 C 源程序可以由一个或多个函数组成。
 - (2) ☒ C 源程序的每行只能写一条语句。
 - (3) ☒ 在 C 源程序中，main 函数必须位于程序的最前面。
 - (4) ☒ C 语言本身没有输入输出语句。
 - (5) ☒ 在 C 源程序中，注释说明只能位于一条语句的后面。
 - (6) ☒ 一个 C 源程序必须包含一个 main 函数。