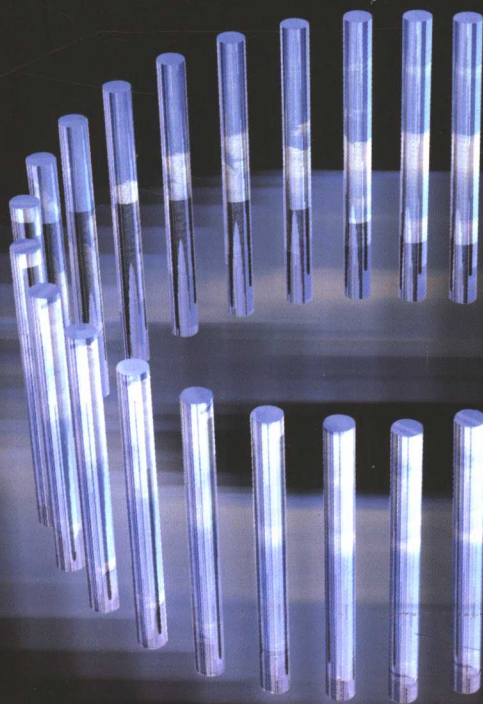


标准 C 程序设计

(第 3 版)

E Balagurusamy 著

金名 张长富 等 译



PROGRAMMING IN ANSI C

Third Edition

清华大学出版社

世界著名计算机教材精选

标准C程序设计

(第3版)

E Balagurusamy 著
金名 张长富 等译

清华大学出版社
北京

E Balagurusamy
Programming in ANSI C, 3e
EISBN:0-07-053477-2

Copyright © 2004 by The McGraw-Hill Companies, Inc.

Original language published by The McGraw-Hill Companies, Inc. All Rights reserved. No part of this publication may be reproduced or distributed by any means, or stored in a database or retrieval system, without the prior written permission of the publisher.

Simplified Chinese translation edition is published and distributed exclusively by Tsinghua University Press under the authorization by McGraw-Hill Education(Asia)Co., within the territory of the People's Republic of China only, excluding Hong Kong, Macao SAR and Taiwan. Unauthorized export of this edition is a violation of the Copyright Act. Violation of this Law is subject to Civil and Criminal Penalties.

本书中文简体字翻译版由美国麦格劳-希尔教育出版(亚洲)公司授权清华大学出版社在中华人民共和国境内(不包括中国香港、澳门特别行政区和中国台湾)独家出版发行。未经许可之出口,视为违反著作权法,将受法律之制裁。未经出版者预先书面许可,不得以任何方式复制或抄袭本书的任何部分。

北京市版权局著作权合同登记号 图字:01-2006-2135号

版权所有,翻印必究。举报电话:010-62782989 13501256678 13801310933

本书封面贴有 McGraw-Hill 公司防伪标签,无标签者不得销售。

图书在版编目(CIP)数据

标准C程序设计(第3版)/(印)巴拉古鲁萨米(Balagurusamy, E.)著;金名、张长富等译. —北京:清华大学出版社, 2006.5

(世界著名计算机教材精选)

ISBN 7-302-12756-5

I. 标… II. ①巴… ②金… ③张… III. C语言—程序设计—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2006)第 025964 号

出版者:清华大学出版社 地址:北京清华大学学研大厦
http://www.tup.com.cn 邮编:100084
社总机:010-62770175 客户服务:010-62776969

责任编辑:龙啟铭

印装者:北京国马印刷厂

发行者:新华书店总店北京发行所

开本:185×260 印张:29.75 字数:737千字

版次:2006年5月第1版 2006年5月第1次印刷

书号:ISBN 7-302-12756-5/TP·8131

印数:1~3000

定价:59.00元

译者序

C 语言是所有本科生课程的一门核心课，也是现今使用最广泛的计算机语言。国内国外的 C 语言图书已经非常多了，但通过本书的翻译，我们觉得这本书还是很有引进价值，具体表现在它的以下几个特点上：

- 本书是基于最新的 C 语言标准。
- 在本书的最后给出了一个完整的应用程序开发示例 PHONE BOOK。
- 扩展讨论了 C 的指针。
- 每章后面的“谨记”一节给出了很有用的编程提示以及可能容易出错的问题。
- 每章后面的案例学习给出了 24 个真实的开发，展示了 C 程序的设计过程。
- 87 个程序设计范例向读者阐述了良好程序设计的基本原则。
- 还有 185 个练习题和 133 道实践编程题。

总之，本书的语言简洁易懂，示例非常丰富且具有很强的实际指导意义，是一本很好的 C 程序设计的教材。

本书的第 2 版被印度的很多学院和大学用作教材。在过去的几年中，本书还被一些重要的软件培训与开发公司用作培训教材。

由于本书的独特性、与众不同的学习方法以及简明的写作风格，相信本书也会受到国内学生和老师的欢迎。

本书由金名、张长富、冯华君、刘守燕、杨咏梅等译，张国清、潘华等人也参与了部分翻译和校对工作。欢迎广大读者指正。

第 1 版前言

C 是一种通用的结构化程序设计语言，它功能强大、高效且简洁。C 集成了高级语言的特性和汇编语言的某些元素，因此，对程序员和计算机都很贴切。用 C 语言进行程序设计很流行，也很有趣。

本书旨在教授读者如何使用 C 语言进行程序设计，不需要读者具有该语句的预备知识，适合于初学者和有经验的程序员。

全书贯彻了“用示例学习”的概念。在深入介绍了 C 语言的每个特性之后，给出了一个完整的示例程序，用于演示说明其应用。每章末尾的“案例学习”不仅介绍了把 C 语言的特性集成在一起的常用方法，而且还显示了它在实际生活中的应用。只要有必要，就用图形来描述，以便于读者更好地理解。最后一章介绍了开发高效、无错误的 C 程序的一些指导原则。

本书包含了 100 多个示例和程序。所有程序都使用与 UNIX 和 MS-DOS 操作系统兼容的编译器进行了测试，并适当地讨论了输出结果。这些程序还演示了良好编程风格的一般原则。

第 3 版前言

作为使用最广泛的计算机语言，C 是所有本科生课程的一门核心课。本书的第 2 版被很多学院和大学用作教材。在过去的几年中，本书还被一些重要的软件培训与开发公司用作培训教材。由于本书的独特性、与众不同的学习方法以及简明的写作风格，本书在学生和老师中很受欢迎。

本书融入了过去 10 年中采用本书的学生和老师的很多反馈意见。本版的特点包括：

- 根据读者的反馈，全书的内容和程序都做了重新评阅和修改。
 - 每章末尾的练习分成了两部分，即“复习题”和“编程练习”。
 - 每章都添加了针对本章内容的大量问答题和问题求解。
 - 在很多章中添加了“谨记”一节。该节列举出了一些有用的提示和容易出现的问题。
 - 新加的附录Ⅲ“电话簿程序”，有助于读者全面体验 C 语言的特性。
- 感谢给本书提供建议的老师和学生，这些建议极大地提高了本书的质量。

E Balagurusamy

目 录

第 1 章 C 语言概述	1	2.7.1 整数型	28
1.1 C 语言的历史	1	2.7.2 浮点数类型	29
1.2 C 语言的主要特性	2	2.7.3 void 类型	30
1.3 示例程序 1: 显示一条消息	3	2.7.4 字符类型	30
1.4 示例程序 2: 两个数相加	6	2.8 变量的声明	30
1.5 示例程序 3: 利息计算	7	2.8.1 基本类型的声明	30
1.6 示例程序 4: 子例程的使用	9	2.8.2 自定义类型的声明	32
1.7 示例程序 5: 数学函数的使用	10	2.9 存储类的声明	33
1.8 C 程序的基本结构	12	2.10 变量的赋值	34
1.9 编程风格	13	2.10.1 赋值语句	35
1.10 运行一个程序	13	2.10.2 从键盘读取数据	37
1.11 UNIX 系统环境下	14	2.11 符号常量的定义	40
1.11.1 创建程序	14	2.11.1 可修改性	40
1.11.2 编译与链接	15	2.11.2 可理解性	40
1.11.3 运行程序	15	2.12 将变量声明为常量	41
1.11.4 创建自己的可运行文件	16	2.13 将变量声明为可变的	41
1.11.5 多个源文件问题	16	2.14 数据的溢出	42
1.12 MS-DOS 系统环境下	17	2.15 案例学习	43
1.13 复习题	18	2.15.1 平均数计算	43
1.14 编程练习	19	2.15.2 温度转换问题	44
第 2 章 常量、变量及数据类型	21	2.16 复习题	45
2.1 概述	21	2.17 编程练习	47
2.2 字符集	21	第 3 章 运算符与表达式	48
2.3 C 标记符	22	3.1 概述	48
2.4 关键字与标识符	22	3.2 算术运算符	48
2.5 常量	23	3.2.1 整数算术运算	49
2.5.1 整型常量	24	3.2.2 实数算术运算	50
2.5.2 实数常量	25	3.2.3 混合算术运算	50
2.5.3 单字符常量	25	3.3 关系运算符	50
2.5.4 字符串常量	26	3.4 逻辑运算符	52
2.5.5 反斜杠字符常量	26	3.5 赋值运算符	52
2.6 变量	26	3.6 递增和递减运算符	54
2.7 数据类型	27	3.7 条件运算符	55

3.8 逐位运算符	56	4.6.1 库存报告	95
3.9 特殊运算符	56	4.6.2 可靠性图形	97
3.9.1 逗号运算符	56	4.7 复习题	99
3.9.2 sizeof 运算符	57	4.8 编程练习	101
3.10 算术表达式	58	第 5 章 判断与分支	102
3.11 表达式的计算	58	5.1 概述	102
3.12 算术表达式的优先级	59	5.2 if 判断语句	102
3.13 一些可计算性问题	61	5.3 简单 if 语句	103
3.14 表达式中的类型转换	62	5.4 if...else 语句	106
3.14.1 隐式类型转换	62	5.5 嵌套 if...else 语句	109
3.14.2 显示转换	64	5.6 阶梯式 else if 语句	112
3.15 运算符的优先级及其关联	66	5.7 switch 语句	116
3.16 数学函数	67	5.8 ?: 运算符	119
3.17 案例学习	68	5.9 goto 语句	122
3.17.1 销售人员的工资	68	5.10 案例学习	125
3.17.2 二次方程的求解	69	5.10.1 数值的分布范围	125
3.18 复习题	71	5.10.2 账单计算	127
3.19 编程练习	73	5.11 复习题	130
第 4 章 输入输出操作管理	75	5.12 编程练习	132
4.1 概述	75	第 6 章 判断与循环	135
4.2 读取一个字符	75	6.1 概述	135
4.3 写字符	78	6.2 while 语句	137
4.4 格式化输入	79	6.3 do 语句	139
4.4.1 整数输入	80	6.4 for 语句	141
4.4.2 实数输入	82	6.4.1 简单的 for 循环语句	141
4.4.3 字符串输入	83	6.4.2 for 循环的其他特性	144
4.4.4 混合数据类型的读取	85	6.4.3 for 循环的嵌套	146
4.4.5 错误输入的检测	85	6.5 循环中的跳转	148
4.4.6 使用 scanf 函数时应记 住的几个要点	87	6.5.1 跳出循环	148
4.5 格式化输出	88	6.5.2 跳过循环的一部分	152
4.5.1 整数的输出	89	6.5.3 避免使用 goto 语句	155
4.5.2 实数的输出	90	6.5.4 简洁的测试表达式	155
4.5.3 单个字符的显示	92	6.6 案例学习	156
4.5.4 字符串的显示	92	6.6.1 二项式系数表	156
4.5.5 混合数据的输出	93	6.6.2 柱状图	158
4.5.6 提高输出的可读性	94	6.6.3 最小成本	160
4.6 案例学习	95	6.6.4 描绘两函数的曲线图	161

6.7 复习题	163	8.8.4 strlen()函数	223
6.8 编程练习	166	8.8.5 其他字符串函数	224
第7章 数组	169	8.9 字符串表	226
7.1 概述	169	8.10 字符串的其他特性	228
7.2 一维数组	170	8.11 案例学习	229
7.3 一维数组的声明	171	8.11.1 计算文本中的字数	229
7.4 一维数组的初始化	173	8.11.2 客户列表处理程序	231
7.4.1 编译时初始化	174	8.12 复习题	233
7.4.2 运行时初始化	175	8.13 编程练习	235
7.5 二维数组	177	第9章 自定义函数	237
7.6 二维数组的初始化	181	9.1 概述	237
7.7 多维数组	185	9.2 为什么需要自定义函数	237
7.8 动态数组	186	9.3 多函数程序	238
7.9 数组的其他	186	9.4 自定义函数的元素	240
7.10 案例学习	188	9.5 函数的定义	241
7.10.1 数列的中值问题	188	9.5.1 函数头	241
7.10.2 标准偏差的计算	190	9.5.2 函数名与类型	241
7.10.3 测试评分	192	9.5.3 形参列表	242
7.10.4 产品与销售分析	194	9.5.4 函数体	242
7.11 复习题	201	9.6 返回值及其类型	243
7.12 编程练习	203	9.7 函数调用	244
第8章 字符数组与字符串	206	9.8 函数声明	246
8.1 概述	206	9.9 函数的类型	248
8.2 字符串变量的声明与初始化	206	9.10 无参数无返回值的函数	248
8.3 使用 scanf 函数从终端读取字符串	208	9.11 有参数无返回值的函数	250
8.3.1 读取文本行	210	9.12 有参数有返回值的函数	253
8.3.2 使用 getchar 和 gets 函数	210	9.13 无参数但有一个返回值的函数	257
8.4 在屏幕上显示字符串	213	9.14 返回多个值的函数	257
8.4.1 使用 printf 函数	213	9.15 函数的嵌套	259
8.4.2 使用 putchar 和 puts 函数	216	9.16 函数的迭代	260
8.5 字符的算术运算	217	9.17 将数组传递给函数	261
8.6 将字符串组合在一起	219	9.17.1 一维数组	261
8.7 两个字符串的比较	220	9.17.2 二维数组	265
8.8 字符串处理函数	221	9.18 将字符串传递给函数	266
8.8.1 strcat()函数	221	9.19 变量的作用域、可见性和生存期	267
8.8.2 strcmp()函数	222	9.19.1 自动变量	267
8.8.3 strcpy()函数	222	9.19.2 外部变量	269
		9.19.3 外部声明	271

9.19.4 静态变量	273	11.12 指针数组	335
9.19.5 寄存器变量	274	11.13 指针作为函数的参数	336
9.19.6 嵌套代码块	275	11.14 函数返回指针	339
9.20 多文件程序	276	11.15 指向函数的指针	340
9.21 案例学习	279	11.16 指针与结构	342
9.22 复习题	282	11.17 案例学习	346
9.23 编程练习	286	11.17.1 考试成绩处理程序	346
		11.17.2 库存更新程序	349
第 10 章 结构与联合	288	11.18 复习题	351
10.1 概述	288	11.19 编程练习	352
10.2 结构的定义	288		
10.3 声明结构变量	289	第 12 章 文件管理	354
10.4 访问结构成员	291	12.1 概述	354
10.5 结构的初始化	292	12.2 定义并打开文件	355
10.6 结构变量的复制与比较	294	12.3 关闭文件	356
10.7 单个成员的运算	295	12.4 文件的输入输出操作	357
10.8 结构数组	296	12.4.1 getc 与 putc 函数	357
10.9 结构中的数组	298	12.4.2 getw 和 putw 函数	358
10.10 结构中的结构	300	12.4.3 fprintf 与 fscanf 函数	360
10.11 结构与函数	302	12.5 I/O 操作的错误处理	362
10.12 联合	304	12.6 随机访问文件	365
10.13 结构的大小	306	12.7 命令行参数	369
10.14 位域	306	12.8 复习题	372
10.15 案例学习	309	12.9 编程练习	373
10.16 复习题	313		
10.17 编程练习	316	第 13 章 动态内存分配与链表	374
第 11 章 指针	319	13.1 概述	374
11.1 概述	319	13.2 动态内存分配	374
11.2 理解指针	319	13.3 用 malloc 函数分配一块内存	375
11.3 访问变量的地址	321	13.4 用 calloc 函数分配多个内存块	377
11.4 指针变量的声明	322	13.5 用 free 函数释放已用的空间	378
11.5 指针变量的初始化	323	13.6 用 realloc 函数改变内存块的大小	378
11.6 通过指针访问变量	325	13.7 链表的概念	380
11.7 指针链	327	13.8 链表的优点	382
11.8 指针表达式	328	13.9 链表的种类	383
11.9 指针的递增与比例因子	329	13.10 再论指针	384
11.10 指针与数组	330	13.11 创建链表	386
11.11 指针与字符串	333	13.12 插入一个数据项	389
		13.13 删除一个数据项	392

13.14 链表的应用	394	15.3.3 输入/输出格式	421
13.15 案例学习	396	15.3.4 程序的通用性	421
13.15.1 在已排序链表中 插入数据	396	15.4 常见的程序错误	422
13.15.2 构建一个已排序的链表	399	15.4.1 丢失分号	422
13.16 复习题	402	15.4.2 误用分号	422
13.17 编程练习	403	15.4.3 丢失括号	423
第 14 章 预处理器	405	15.4.4 丢失引号	424
14.1 概述	405	15.4.5 误用引号	424
14.2 宏替换指令	405	15.4.6 使用不正确的注释字符	424
14.2.1 简单宏替换	406	15.4.7 未定义变量	425
14.2.2 含参数的宏	408	15.4.8 忽视了运算符的优先级	425
14.2.3 宏嵌套	409	15.4.9 忽视了递增递减运算符 的计算顺序	426
14.2.4 文件包含	410	15.4.10 忽视了函数参数的说明	426
14.4 编译器控制指令	410	15.4.11 在函数调用中实参和 形参类型不匹配	426
14.4.1 情形 1	411	15.4.12 函数未声明	427
14.4.2 情形 2	412	15.4.13 在 scanf 的参数中丢失了 &运算符	427
14.4.3 情形 3	412	15.4.14 超出了数组的边界	428
14.4.4 情形 4	413	15.4.15 忘记了给字符串的空字符 留出空间	428
14.5 ANSIC 的其他预处理器指令	414	15.4.16 使用未初始化的指针	428
14.5.1 #elif 指令	414	15.4.17 丢失了间接运算符和 地址运算符	428
14.5.2 #pragma 指令	414	15.4.18 在指针表达式中丢失 括号	429
14.5.3 #error 指令	415	15.4.19 在宏定义语句中参数 遗漏了括号	429
14.5.4 字符串化运算符	415	15.5 程序测试与调试	430
14.5.5 标记符粘贴运算符	416	15.5.1 错误的类型	430
14.6 复习题	416	15.5.2 程序测试	431
14.7 编程练习	417	15.5.3 程序调试	432
第 15 章 C 程序开发指导原则	418	15.6 程序的效率	432
15.1 概述	418	15.6.1 运行时间	432
15.2 程序设计	418	15.6.2 内存需求	432
15.2.1 问题分析	418	15.7 复习题	433
15.2.2 勾勒程序结构	418		
15.2.3 算法开发	419		
15.2.4 控制结构	419		
15.3 程序编码	420		
15.3.1 自身文档化	420		
15.3.2 语句构造	421		
		附录 I 位级程序设计	435

I.1 概述.....	435	I.5 屏蔽.....	439
I.2 逐位逻辑运算符.....	435	附录 II.....	440
I.2.1 逐位与操作.....	435	附录 III.....	441
I.2.2 逐位或操作.....	437	附录 IV 电话簿示例程序.....	444
I.2.3 逐位非或操作.....	437		
I.3 逐位移位运算符.....	438		
I.4 逐位求反运算符.....	438		

第 1 章 C 语言概述

1.1 C 语言的历史

作为一种程序设计语言，字母“C”看上去是一个奇怪的名字。但是这个奇怪而好听的语言却是现今最为流行的计算机语言之一，因为它是一种结构化的、高级的、与机器无关的语言。它允许软件开发人员开发程序时无须担心实现这些程序的硬件平台。

所有现代语言的起源都是 ALGOL 语言，该语言是 20 世纪 60 年代引入的。ALGOL 语言是最先使用块结构的计算机语言。尽管它从来没有在美国流行开来，但在欧洲被广泛使用。ALGOL 语言给计算机科学界带来了结构化程序设计的概念。20 世纪 60 年代，计算机科学家，如 Corrado Bohm、Guiseppe Jacopini 和 Edsger Dijkstra 使这一概念大众化了。随后，又宣布开发了好几种计算机语言。

1967 年，Martin Richards 开发了一种称为 BCPL（基本组合程序设计语言）的计算机语言。该语言主要用于系统软件的开发。1970 年，Ken Thompson 创建了一种计算机语言，该语言继承了 BCPL 的很多特性，且就称为 B 语言。在贝尔实验室，B 语言用来开发 UNIX 操作系统的早期版本。BCPL 和 B 语言都是“无类型”的系统程序设计语言。

C 语言是 Dennis Ritchie 于 1972 年在贝尔实验室从 ALGOL、BCPL 和 B 语言的基础上发展而来的。C 语言利用了这些语言的很多概念，并添加了数据类型的概念以及其他功能强大的特性。由于它是与 UNIX 操作系统一起被开发出来的，因此它与 UNIX 有着很强的关联。UNIX 操作系统（也是在贝尔实验室开发出来的）几乎完全是用 C 语言编码的。UNIX 是现今使用最为流行的网络操作系统，也是因特网数据超高速公路的心脏。

多年以来，C 语言主要用于科研环境下，但最终，随着多种商用 C 编译器的发布，以及 UNIX 操作系统的不断流行，在计算机专业中也开始获得广泛支持。今天，C 语言可以运行在多种操作系统和硬件平台下。

20 世纪 70 年代，C 语言发展为现在所谓的“传统 C 语言”。自 1978 年由 Brian Kerningham 和 Dennis Ritchie 著作的 *The C Programming Language* 一书的出版，C 语言成了最为流行的语言。该书很受欢迎，以至于在程序设计界，C 语言就认为是“K&R C”。C 语言的快速发展导致了不同版本的语言出现，这些语言类似但往往不兼容。对系统开发人员来说，这是一个严重的问题。

为了确保 C 语言的标准，1983 年，美国国家标准局（American National Standards Institute, ANSI）任命了一个技术委员会来定义 C 语言的标准。该委员会于 1989 年批准了一个 C 语言版本，这就是现在的 ANSI C。该本版又于 1990 年被国际标准组织（International Standards Organisation, ISO）批准。该标准于 1999 年进行了更新。C 语言的历史如图 1.1 所示。

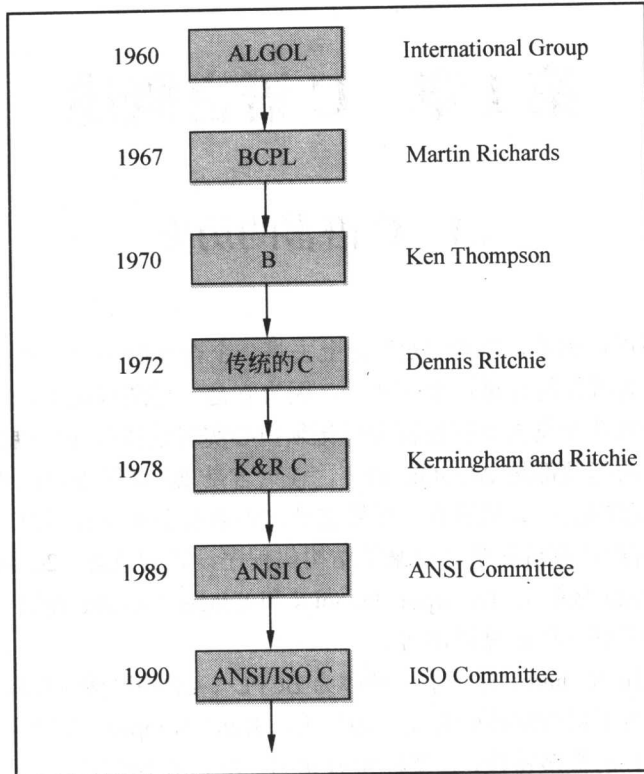


图 1.1 C 语言的历史

1.2 C 语言的主要特性

C 语言之所以越来越受欢迎,是因为它具有很多可取的特性。它是一种健壮的语言,其丰富的内置函数集和运算符可用来编写任意复杂的程序。C 语言编译器融合了汇编语言的性能和高级语言的特性,因此很适合编写系统软件和商业软件。事实上,市场上很多的 C 编译器都是用 C 语言编写的。

用 C 语言编写的程序高效且运行速度快。这归功于它的各种数据类型和功能强大的运算符。其运行速度是 BASIC 语言的好几倍。例如,一个计算 0~1500 的程序,用 C 语言编写,其运行时间为 1 秒,而用 BASIC 语言编写的则要运行 50 秒。

C 语言有 32 个关键字,其长度取决于其内置函数。C 语言有很多标准函数可用来开发程序。

C 语言是高度可移植的。这意味着,为某一台计算机编写的 C 程序,只要稍作修改甚至不用修改就可以在另一台计算机上运行。如果我们计划使用不同操作系统的新计算机时,可移植性很重要。

C 语言很适于结构化程序设计,因而要求用户以功能模块的方式来思考问题。这些模块的恰当结合就可以形成一个完整的程序。这种模块化的结构使得程序的调试、测试和维

护更加容易。

C 语言另一个重要特性是它的自我扩展能力。一个 C 程序基本上是各种函数的集合，这些函数由 C 函数库支持。我们可以不断地将自己的函数添加到 C 函数库中去。由于有了这么大量的函数，程序设计就变得简单了。

在讨论 C 语言的具体特征之前，我们先来看看一个 C 范例程序，分析并了解程序是如何工作的。

1.3 示例程序 1：显示一条消息

请看图 1.2 所示的一个很简单的程序。

```
main( )
{
/*.....printing begins.....*/

    printf("I see, I remember");

    /*.....printing ends.....*/
}
```

图 1.2 显示一行文字的程序

该程序将产生如下的输出：

```
I see, I remember.
```

让我们来详细看看该程序。第一行告诉操作系统程序名是 `main`，从这一行开始执行。`main()` 函数是 C 系统使用的一个特殊函数，用来告诉计算机程序的运行起点。每个程序必须正好有一个 `main` 函数。如果有多个 `main` 函数，编译器就无法知道哪一个是程序。紧跟在 `main` 后面的空括号对表明 `main` 函数不带参数。关于参数的概念我们将在讨论函数（第 9 章）时再详细介绍。

第二行的开始是“{”，表明 `main` 函数的开始，而最后一行的闭括号则表示该函数的结束。在上面范例程序中，闭括号还标志着整个程序的结束。这两个括号之中的所有语句就形成了函数体。函数体包含有一个指令集，从而完成指定的任务。

在上面的范例程序中，函数体包含有 3 个语句，其中只有 `printf` 一行是可执行的语句。以 `/*` 开始，以 `*/` 结尾的行称为注释行。程序中恰当地使用注释行，可以提高程序的可读性。更容易让人理解。由于注释行不是可执行语句，因此 `/*` 与 `*/` 之间的内容全部被编译器忽略掉。通常，一个注释可以插入到程序的任何空白处——行的开始、中间或结尾处——但不能插入到一个词的中间。

尽管注释行可以出现在程序的任意地方，但在 C 语言中它们不能嵌套。这就意味着，不能在注释行中再插入注释行。一旦编译器发现了注释的开始标志，它就将忽略掉后面的

所有内容，直到再发现一个结束标志为止。下面注释行：

```
/* = = /* = = */ = = */
```

是不合法的，因此将产生一个错误。

由于注释行不会影响程序的运行速度以及编译后程序的大小，因此我们应大方地在程序中使用注释。注释有助于开发人员和其他用户理解程序的不同函数和运算，对程序的调试和测试也有帮助。我们将在后面的范例程序中体会到注释行的作用。

现在让我们来看 `printf()` 函数，这是该范例程序中唯一可执行的语句：

```
printf("I see, I remember! ");
```

`printf` 是预定义的标准 C 函数，用于显示输出。预定义的含义就是，该函数已编写好并已编译。在链接时，与我们的程序链接在一起。编译和链接的概念将在本章的后面介绍。`printf` 函数将两个引号之间的内容显示出来。在本范例程序中，其输出为：

```
I see, I remember!
```

注意，打印行以分号结尾。C 语言的每条语句都必须以分号结尾。

假如我们要如下地将输出显示成两行：

```
I see,  
I remember!
```

这可以通过添加两个 `print` 函数来实现，具体如下：

```
printf("I see, \n ");  
printf("I remember! ");
```

两个括号之间的信息称为该函数的参数。第一个 `printf` 函数的参数是 “I see, \n”，第二个的是 “I remember!”。这些参数只是要显示出来的字符串。

注意，第一个 `printf` 函数的参数在字符串的末尾包含有字符 `\` 和 `n` 的组合。该组合称为新行字符。新行字符命令计算机换到下一行（即新行）。在概念上，它类似于打字机的回车键。在显示了逗号字符后，新行字符 `\n` 的出现将使 “I remember!” 显示在下一行。字符 `\` 和 `n` 之间不允许有空格。

如果省略掉第一个 `printf` 语句的新行字符，那么输出仍为一行，即如下所示：

```
I see,I remember!
```

这类似于图 1.2 中的程序的输出。但是，注意，在 “,” 和 “I” 之间没有空格。

只使用一个 `printf` 语句，只要在适当的地方使用新行字符，也可以生成两行或多行输出。例如，语句：

```
printf("I see, \n I remember! ");
```

的输出为：

```
I see,  
I remember!
```

而语句：


```
printf("I \n see, \n I\n remember! ");
I
    see,
        I
            remember!
```

注意，有些作者推荐在所有程序的开头加上如下的输入输出库函数：

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

但是，对 `printf` 和 `scanf` 并没有必要，因为这两个函数已经定义为 C 语言的一部分了。关于输入输出函数的更多内容请参见第 4 章。

在我们继续讨论其他更多的示例之前，必须注意到很重要的一点：C 语言是区分大小写字母的。例如，`printf` 和 `PRINTF` 并不相同。在 C 语言中，通常都是写成小写字母。而大写字母用作表示常量的符号名。我们也可以在输出字符串中使用大写字母，例如，“I SEE” 和 “I REMEMBER”。

上面介绍的用于显示 “I see,I remember!” 的示例是最简单的程序之一。图 1.3 列举了这类简单程序的一般格式。所有 C 程序都需要一个 `main` 函数。

main 函数

Main 函数是每个 C 程序的一部分。C 语言允许有如下多种形式的 `main` 函数声明：

```
main()
```

```
int main()
```

```
void main()
```

```
main(void)
```

```
void main(void)
```

```
int main(void)
```

空括号对表示该函数不带参数。这也可以在括号中使用关键字 `void` 来明确表示不带参数。我们还可以在 `main` 之前指定 `int` 或 `void`。关键字 `void` 表示该函数不给操作系统返回任何信息，而 `int` 表示函数将返回一个整数值给操作系统。如果指定为 `int`，那么程序的最后一行必须是 “`return 0;`”。为了简单起见，我们在本书的所有示例程序中使用第一种形式的 `main` 函数。

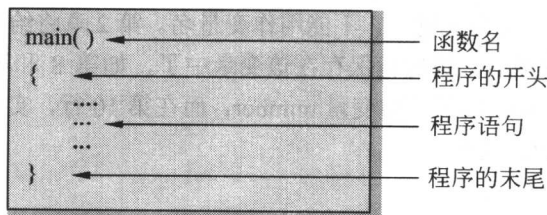


图 1.3 简单程序的结构